Universidade de Aveiro
2008

Departamento de Electrónica
Telecomunicações e Informática

Hugo José Mostardinha de Almeida

Desenvolvimento dum software para um sistema de medição de antenas

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Doutor José Fernando da Rocha Pereira, Professor Associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri

Presidente	Doutor Tomás António Mendes Oliveira e Silva Professor Associado da Universidade de Aveiro
Vogais	
	Doutor José Fernando da Rocha Pereira (Orientador) Professor Associado da Universidade de Aveiro
	Doutor Artur Manuel Oliveira Andrade de Moura Professor Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

agradecimentos

É com uma enorme satisfação que aproveito este espaço para expressar algumas palavras para com as pessoas que, das formas mais diversas, me ajudaram ao longo desta dissertação.

Um sincero obrigado e profunda estima para o meu Orientador, Professor Doutor José Rocha Pereira, por todo o apoio científico e pedagógico, bem como por toda a sua colaboração ao longo desta dissertação.

Agradeço ao Professor Doutor Bernardo Cunha, Professor Auxiliar da Universidade de Aveiro, pela orientação na linguagem de programação e conselhos de utilização.

Um grande obrigado ao Eng.º Pedro Almeida do IEETA - Aveiro, pela partilha de conhecimento e experiência no mundo da linguagem de programação C#.

Um forte abraço de agradecimento ao meu "primo", Doutor António Neves do IEETA - Aveiro, pela sua disponibilidade, pelas palavras de conforto e motivação que me ajudaram a chegar à fase final deste projecto de vida.

Agradeço ao Eng.º Pedro do Mar do IT pelas suas opiniões e conselhos, principalmente pela sua disponibilidade para testar o RadScan e dar o seu parecer.

Agradeço igualmente a todas a pessoas que se cruzaram comigo e que de algum modo me transmitiram conhecimentos do âmbito das linguagens de programação e da radiofrequência, permitindo a redacção da presente dissertação.

Um forte abraço de agradecimento ao meu irmão, Eng.º Pedro Mostardinha, pela sua enorme paciência, dedicação, pelo auxílio que me prestou na resolução de entraves na programação e na tradução e por todas as suas sugestões.

Um eterno abraço de agradecimento à minha noiva, Professora Paula Cardoso, pelo seu inteiro apoio e motivação, pela compreensão nos momentos de ausência e pelas privações.

À minha família, agradeço a compreensão pelos momentos de ausência, as palavras de incentivo e a partilha de todas as alegrias e tristezas.

Finalmente, dedico esta dissertação à minha Mãe, pelos seus inúmeros sacrifícios, ajudando-me a ultrapassar as dificuldades e incentivando-me sempre, especialmente nos momentos mais difíceis da minha vida pessoal, académica e profissional.

palavras-chave

Diagramas de radiação de antenas, câmara anecoica, GPIB, linguagem C#, usabilidade de *software*

O trabalho apresentado nesta dissertação descreve o desenvolvimento de um *software* para Windows, que tem por missão gerir o sistema de medição de diagramas de antenas, associado à câmara anecóica instalada no Departamento de Electrónica, Telecomunicações e Informática (DETI) da Universidade de Aveiro (UA).

Este *software* visa substituir uma versão desenvolvida anteriormente para ambiente MS-DOS. Para além das funcionalidades de controlo, esta versão dá uma acentuada importância à interface com o utilizador. À semelhança dos sistemas operativos e aplicações actuais e vindouras, que revelam cada vez maior facilidade de utilização, exige-se deste *software* um grande cuidado na sua implementação para que seja fácil de utilizar (*user friendly*), bem como abranja um universo mais amplo de utilizadores. Neste sentido, utiliza-se no âmbito desta dissertação ferramentas de avaliação da usabilidade das interfaces humano-computador.

Por fim, perspectiva-se também o futuro do sistema ao nível de funções a desenvolver, tais como novas formas de comunicar com dispositivos mais recentes.

keywords

antenna radiation paterns, anechoic chamber, GPIB, C# programming language, software usage.

The work presented in this dissertation describes the development of a windows software tool to manage the antenna pattern measurement system, associated with an anechoic chamber installed in the electronic and telecommunications department (DETI) in the University of Aveiro.
The purpose of this tool is to replace a previous tool designed for MS-DOS environment. Beyond the control functions, it is focussed on the user interface. Similar to operating systems and to today and tomorrow's applications, that reveal an increasing ease of use, this software was carefully studied to be user-friendly, for a wider range of public. With this in mind, user-pc usage evaluation tools are used. Suggestions for future developments of this tool have been also considered, such as new functions and new ways to communicate with new devices.

Índice

Introdução 1
1 Estudo de Usabilidade 7
1.1 Grupo de Utilizadores 7
1.2 Estudo do Software em MS-DOS 8
1.2.1 Análise das tarefas do software 8
1.2.2 Capacidades e limitações de usabilidade do *software* existente. 8
Menu inicial 9
1.2.3 Posicionamento inicial *(Offset)* 10
1.2.4 Aquisição contínua: 10
1.2.5 Gráficos 11
1.2.6 Gravar e ler ficheiros. 13
1.2.7 Editar Valores das Medidas. 14
1.2.8 Apreciação Geral 15
1.3 Estudo e concepção da Maqueta 15
1.3.1 Identificação de Objectivos de Usabilidade 15
1.3.2 Identificação de Funcionalidades a oferecer pelo sistema 16
1.3.3 Proposta de Modelo Conceptual 18
1.3.4 Plano de Avaliação 19
2 Plataforma de desenvolvimento 27
2.1 Selecção de Linguagem de Programação 28
2.2 Interligação e comunicação entre equipamentos 28
2.2.1 O protocolo GPIB 29
2.2.2 Tipo de mensagens GPIB 29
2.2.3 Controladores, equipamentos emissores e receptores 30
2.2.4 SCPI 31
2.2.5 O modelo de instrumentos SCPI 32
2.2.6 Exemplo de um comando SCPI 34
3 Desenvolvimento do *software* de controlo 35
3.1 Rotinas de comunicação com equipamentos GPIB 35
3.1.1 Ferramentas de Comunicação 36
3.1.2 Comunicação com o Voltímetro 37
3.1.3 Comunicação com o Posicionador 38
3.2 Funções de comunicação com os dispositivos 51
3.3 Desenvolvimento de comando e interface 55
3.3.1 Aquisição de dados e interface com o utilizador 55
3.3.2 Apresentação de dados 59
4 Testes e Resultados 63
4.1 Resultados do estudo de usabilidade 63
4.2 Fiabilidade de valores 63
4.3 Performance do sistema 65
5 Especificações Técnicas 67
5.1.1 Descrição do *hardware* envolvido no sistema: 67
5.1.2 Requisitos do *software* RadScan 68
5.1.3 Características Técnicas do sistema 68
Conclusão 71

Referências 75
Anexo A. - Heurísticas de Usabilidade de Jakob Nielsen 77
Anexo B. - Classificação de erros 79
Anexo C. - GPIB sinais e linhas 81
Anexo D. - Manual do utilizador 93
Anexo E. - Código 95

Índice de Figuras

Figura 1 – Ilustração da câmara anecoica do DETI 3
Figura 2 - Sistema de medição de diagrama de antenas 4
Figura 3 - Menu inicial. 9
Figura 4 - Sequência de menus, para o posicionamento inicial. *(Offset)*. 10
Figura 5 - Sequências de menus para definir parâmetros do varrimento. 10
Figura 6 - Gráfico de radiação em coordenadas lineares. 11
Figura 7 - Gráfico de radiação em coordenadas polares. 12
Figura 8 - Menu Escolher mensagens para o gráfico. 12
Figura 9 - Menu mudar a gama de valores do gráfico. 12
Figura 10 - Menu escolher directório para gravar/ler. 13
Figura 11 - Menu ler valores do disco. 13
Figura 12 - Apresentação de dados do ficheiro lido a partir do disco. 14
Figura 13 - Menu Editar/ver valores das medidas. 14
Figura 14 - RadScan, exemplos de visibilidade do sistema. 20
Figura 15 - RadScan, mensagem de erro. 20
Figura 16 - RadSan, menu e janela de diálogo da aquisição. 21
Figura 17 - RadScan, Janela de diálogo abrir ficheiro. 22
Figura 18- RadScan, Janela de correcção de eixo 23
Figura 19 - RadScan, menu ficheiro e ajuda. 23
Figura 20 - National Instruments Measurement & Automation Explorer. 36
Figura 21 - Janela de comunicação isolada com bus GPIB. 37
Figura 22 - NI Spy, captura 39
Figura 23 - Ícone do Commlink 43
Figura 24 - Menu “File”, comando “Axes” 43
Figura 25 - Janela de configuração de parâmetros do controlador. 44
Figura 26 - Menu “File”, comando “Save to NV memory”. 45
Figura 27 - Menu “Move” e seus comandos. 45
Figura 28 - Movimento “Direct”, exemplo de configuração. 46
Figura 29- Movimento “Track”, exemplo de configuração. 47
Figura 30 - Movimento “Raster”, exemplo de configuração. 48
Figura 31 - Movimento “Sector”, exemplo de configuração. 49
Figura 32 - Movimento “Slew” ,exemplo de configuração 49
Figura 33 - Ilustração de tabelas, janelas e menus do RadScan 59
Figura 34 - Tabela de dados adquiridos. 60
Figura 35 - Exemplo: Resultado de “Exportar Excel” 61

Figura 36 - Diagrama de radiação de uma antena Yagi, obtido aquisição por dois *softwares* distintos. 64
Figura 37 - Gráfico ampliado de um troço do diagrama de radiação da antena Yagi. 65
Figura 38 - Ficha GPIB, respectivas linhas de comunicação. 81
Figura 39 - Cabo GPIB (cada extremidade contém duas fichas de ligação macho e fêmea) 83
Figura 40 - Configuração de ligações entre dispositivos, linear a esquerda e estrela a direita. 84
Figura 41 - Evolução das normas GPIB. 85

Índice de Tabelas

Tabela 1 - Resultados da execução das tarefas em ambiente DOS. 26
Tabela 2 - Resultados da execução das tarefas em ambiente Windows... 26
Tabela 3 - Descrição dos bits relevantes do byte de resposta ao comando X4. .. 42
Tabela 4 - Incremento possível face a velocidade 69
Tabela 5 - IEEE 488.2 *Required and Optional Control Sequences* 88
Tabela 6 - IEEE 488.2 *Required and Optional Control Sequences* 89
Tabela 7 - IEEE 488.2 *Mandatory Common Commands* 91

Índice de Diagramas

Diagrama 1 - Funcionalidades da maqueta RadScan. 17
Diagrama 2 - Modelo de instrumento SCPI. 32
Diagrama 3– Ilustração dos protocolos de comunicação com os dispositivos 35
Diagrama 4 - Interacção entre menus e sub-menus no RadScan. 58
Diagrama 5 - *Status Report Model* 92

Índice de Abreviaturas

ANSI	*American National Standards Institute*
ASCII	*American Standard Code for Information Interchange*
CIC	*Controller In Charge*
CPU	*Central Processing Unit*
dB	decibel
DETI	Departamento de electrónica telecomunicações e Informática
GPIB	General Purpose Interface
HP	Hewlett-Packard
HP-IB	Hewlett-Packard Interface Bus
I/O	Input / Output
ID	Identificação
IEEE	Instituto de Engenheiros Electricistas e Electrónicos
ISA	*Industry Standard Architecture*
ISO	International Organization for standardization
JPG	*Joint Photographic Experts Group*
MS-DOS	*Microsoft Corporation* - D*isk Operating System*
NI	National Instruments
PC	*Personal Computer*
RAM	Random access memory
RF	Rádio Frequência
SCPI	*Standard Commands for Programmable Instruments*
SO	Sistema Operativo
TTL	Transistor - Transistor Logic
UA	Universidade de Aveiro
USB	Universal Serial Bus

Introdução

Com a evolução das tecnologias, o aumento da capacidade de processamento traz novos sistemas operativos e aplicações cada vez mais fáceis de usar (*user friendly*). A tendência é cada vez mais a de se focar no utilizador e gerar aplicações de interface com o equipamento que respondam melhor às suas necessidades. Neste contexto, surgiu a necessidade de realização deste projecto.

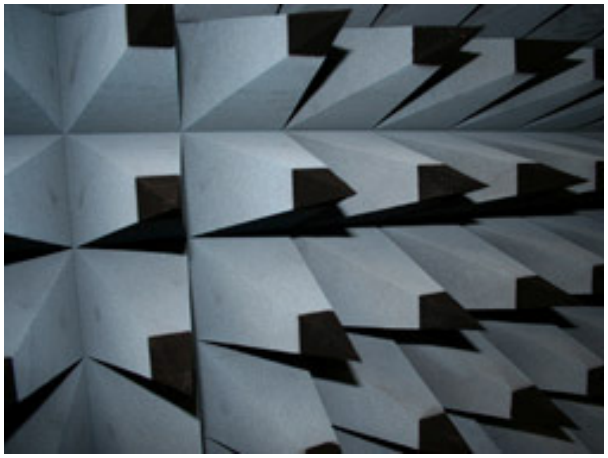
O projecto que a seguir se apresenta consiste no desenvolvimento de um *software* para ser utilizado no sistema de medição de diagramas de radiação de antenas. Diagramas, estes que consistem em gráficos de amplitude de sinal recebido em função da posição angular, permitindo assim ilustrar o modo como a antena radia ou recebe energia electromagnética.

A nova aplicação visa substituir a anterior que é um programa desenvolvido em linguagem C há já vários anos, e que corre num PC com o sistema operativo MS-DOS[1]. Deverá manter as funcionalidades relevantes do *software* anterior, apresentando no entanto um sistema de mais fácil interacção com o utilizador em ambiente de Windows. O desenvolvimento desta aplicação consistirá numa análise e avaliação do software anterior; no desenvolvimento de uma interface com o utilizador, testada através de uma maqueta; desenvolver as rotinas de comunicação para a camada inferior do software e, finalmente com evolução continuada de procedimentos, obter uma versão funcional e intuitiva.

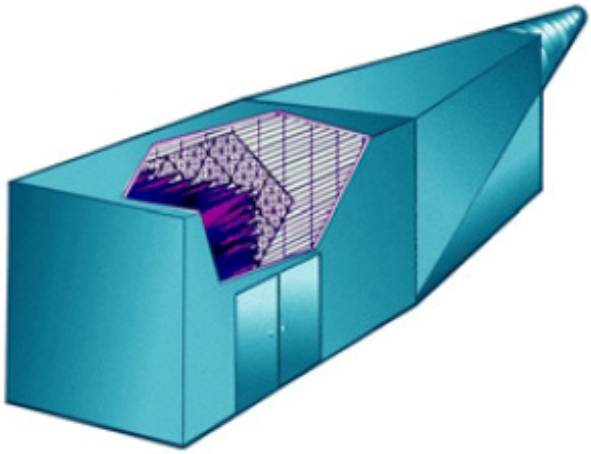
O sistema de medição de diagramas de radiação de antenas tem como ponto de partida uma antena transmissora, que emite um sinal produzido por um

gerador de radiofrequência (RF), com frequência e potência seleccionadas pelo operador. A antena a medir funciona em recepção e é colocada no posicionador que vai permitir a sua rotação em azimute. Durante a rotação, o sinal recebido pela antena a medir será amostrado em amplitude e fase, através de um voltímetro vectorial. Os dados recolhidos permitem a construção de um gráfico de radiação. É fundamental que as amostragens sejam feitas nos mesmos pontos ao longo do eixo de rotação, garantindo deste modo a repetibilidade do processo e possibilitando a combinação de conjuntos de medidas para posterior processamento. Por exemplo, podem combinar-se duas medidas com polarizações ortogonais para obter o diagrama de radiação de uma antena com polarização circular.

Para que o gráfico obtido fosse uma fiel representação do modo como a antena a medir recebe, seria necessário realizar as medições em espaço aberto, sem obstáculos e livre de outras fontes de rádio, isto é, onde só existissem as duas antenas. Como tal situação é hoje em dia muito difícil, ou até impossível de conseguir, utilizam-se como alternativa salas de dimensões apropriadas que emulam o espaço livre. A estas salas dá-se o nome de câmaras anecoicas, por as suas paredes não produzirem ecos. Isto é conseguido através do revestimento das suas paredes com materiais absorventes de radiações electromagnéticas (Figura 1a). As câmaras anecoicas também reduzem as interferências de RF vindas do exterior através da sua blindagem.



a) Revestimento do fundo da câmara



b) Forma da Câmara (*tapered chamber*)

c) Posicionador da antena e parede latera

Figura 1 – Ilustração da câmara anecoica do DETI

Deste sistema de medição faz parte a câmara anecóica do DETI-UA (Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro), (Figura 1). A câmara é do tipo *tapered chamber,* que é composta por três blocos principais: um cone, um tronco de pirâmide e um cubo (Figura 1b). Neste tipo de câmara a antena emissora está colocada no cone e a antena receptora no centro do cubo. Na Figura 1c, podemos ver a antena receptora no posicionador que permite o seu movimento em azimute.

É importante referir também que todo o controlo dos equipamentos de medida e posicionamento é feito usando o protocolo de comunicação GPIB, pelo que no Computador pessoal (PC) está instalada uma placa de interface GPIB.

Na Figura 2, pode ver-se uma representação gráfica do sistema de aquisição de dados e interacção dos dispositivos de medição e controlo[1].

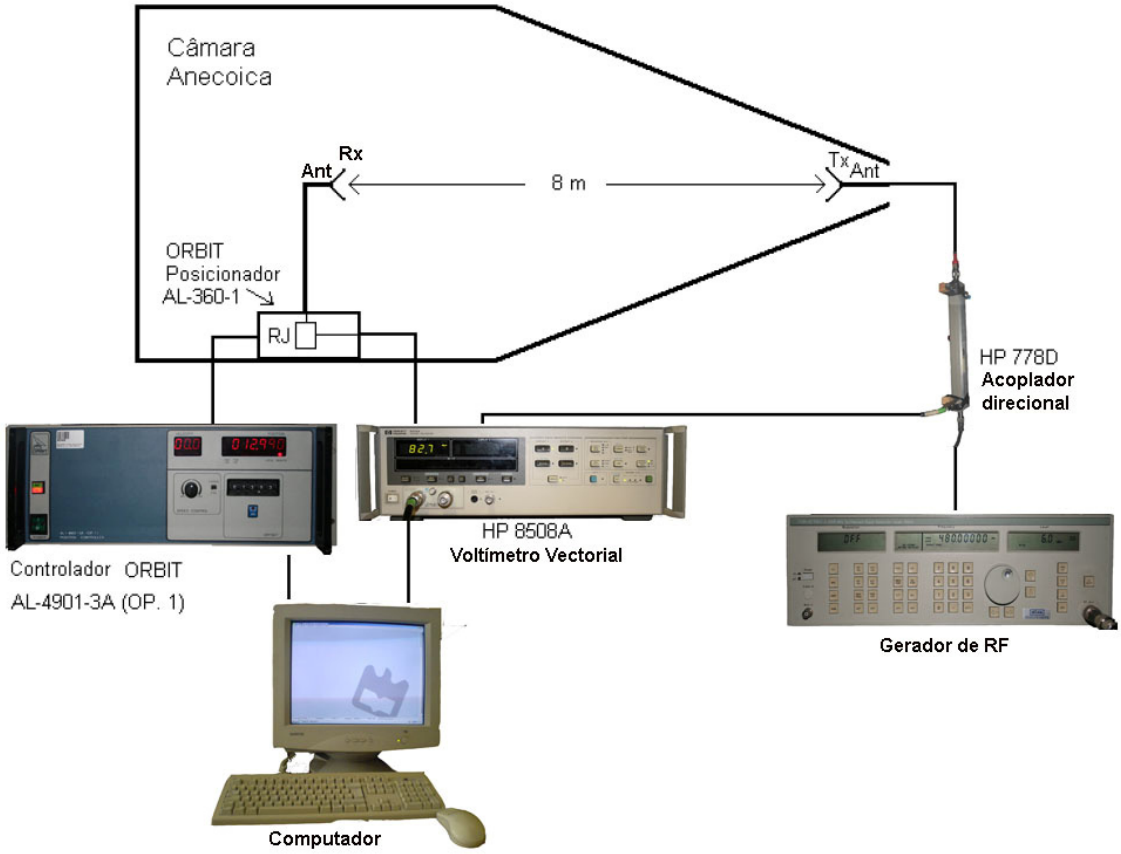


Figura 2 - Sistema de medição de diagrama de antenas

Na presente dissertação de mestrado apresentam-se várias etapas do projecto. Num primeiro momento elaborar-se-á um estudo de usabilidade do *software* existente e desenvolver-se-á uma maqueta para o sujeitar a avaliação.

Posteriormente no Cap. 2 descreve-se a plataforma de desenvolvimento, composta pela linguagem de programação e protocolos de comunicação entre dispositivos.

De seguida no Cap. 3 procede-se à especificação do *software* de controlo, sendo que este capítulo está subdividido em três partes. Na primeira (Comunicação com os equipamentos GPIB) descreve-se a comunicação com o voltímetro vectorial e o controlador do posicionador e ainda as ferramentas usadas. Na segunda, (Funções de comunicação com os dispositivos), faz-se uma

breve discrição das funções de nível mais baixo, responsáveis pela comunicação entre dispositivos e o computador pessoal. Por último, na terceira (Desenvolvimento de comando e interface), analisa-se o processo de aquisição de dados, a interface com o utilizador e os moldes da apresentação de dados. Posteriormente no Cap. 4, descrevem-se os testes e apresentam-se os resultados mais pertinentes para o projecto aqui descrito. De seguida no Cap. 5 descreve-se as características técnicas relevantes do sistema. Finalmente são apresentadas as conclusões gerais sobre o trabalho elaborado no âmbito desta dissertação.

1 Estudo de Usabilidade

Para um programador, por vezes é difícil avaliar as dificuldades dos utilizadores no uso do *software* que desenvolve. Para tentar ultrapassar este obstáculo, usaram-se critérios definidos por especialistas na área da usabilidade[17] para avaliar a usabilidade do *software*, principalmente da sua interface com o utilizador.

Para elaborar uma proposta de *software,* começar-se-á pela análise e avaliação da versão anterior, de modo a melhor compreender, por um lado, e logo à partida, o seu funcionamento; e por outro, as dificuldades inerentes à sua utilização. Desta forma, podem colmatar-se certas lacunas no *software* a desenvolver.

Propõe-se assim a execução de uma maqueta para realizar testes semelhantes aos efectuados na versão antiga do *software* e averiguar a sua usabilidade e satisfação do utilizador, ainda antes da sua concepção. Daqui resultam duas importantes avaliações de usabilidade do *software* existente para MS-DOS e da maqueta entretanto desenvolvida.

1.1 Grupo de Utilizadores

Os actuais grupos de utilizadores são docentes e alunos de projecto do ramo de Electrónica e Telecomunicações. Estes possuem bons conhecimentos de informática na óptica do utilizador e estão familiarizados tanto com as funcionalidades do sistema existente (*hardware* e *software*), como com as suas limitações. Pretende-se atingir um grupo de utilizadores mais abrangente, com experiências e conhecimentos diversificados. Pretende-se nomeadamente alargar o grupo de utilização a alunos universitários de qualquer ano e a docentes de outras áreas.

1.2 Estudo do Software em MS-DOS

1.2.1 Análise das tarefas do software

- Permitir ao utilizador definir a posição central do movimento de rotação (0º).
- Permitir parâmetros da rotação como:
  - Posição inicial e final.
  - Incremento em graus com a frequência com que se pretendem as medidas.
- Apresentação do gráfico de Radiação:
  - Coordenadas polares.
  - Coordenadas lineares.
- Possibilidade de edição dos valores de medidas.
- Guardar e ler medidas num ficheiro compatível com Excel (com extensão .xl).

1.2.2 Capacidades e limitações de usabilidade do *software* existente.

Pretender-se-á analisar o *software* procedendo-se à sequência normalmente executada pelo docente.

Usaremos como critérios de avaliação os princípios de usabilidade segundo Jakob Nielson.[13] (ver anexo A), classificando os erros numa escala de 0 a 4 (ver anexo B).

Menu inicial

No menu inicial (Figura 3), encontram-se todas as tarefas executadas por este *software*:

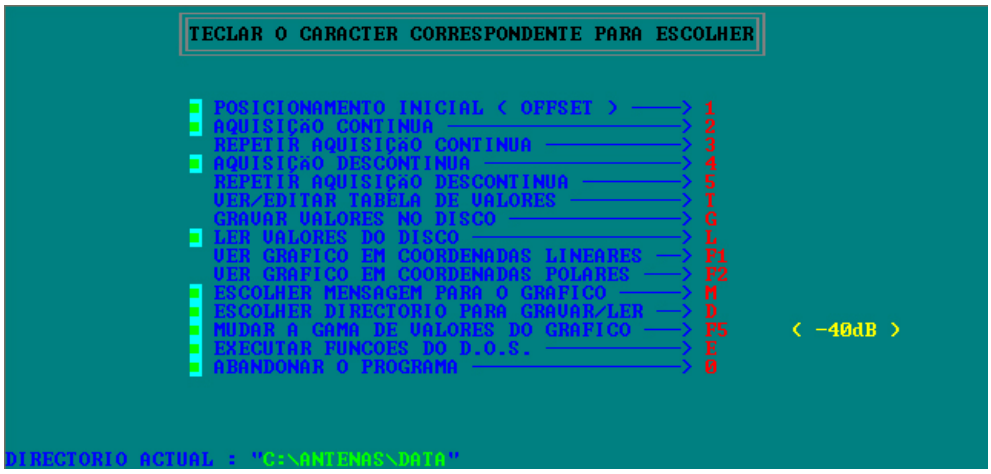


Figura 3 - Menu inicial.

No entanto, observam-se já algumas dificuldades:

1. - A leitura do texto que se apresenta no ecrã é difícil, pois troca-se facilmente o caracter de escolha pelo que está imediatamente acima ou abaixo, pois as cores escolhidas não são as mais adequadas (azul difícil de ler, num fundo verde).

- Violação do ponto 8. “ Design estético e minimalista”, com gravidade 3.

2. - Caso o utilizador queira algum esclarecimento adicional, o *software* é desprovido de qualquer sistema de Ajuda. Não existe um “Manual de Utilização” em formato Papel, existe unicamente o relatório de projecto onde se encontra informação muito técnica, relacionada principalmente com a elaboração do código.

- Violação do ponto 10 “Ajuda e Documentação” com gravidade 1.

1.2.3 Posicionamento inicial *(Offset)*

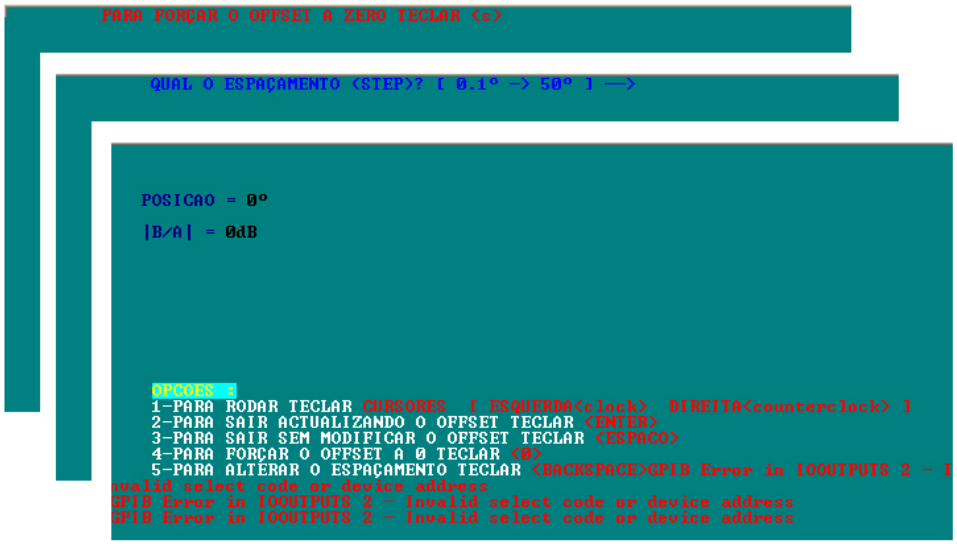


Figura 4 - Sequência de menus, para o posicionamento inicial. *(Offset).*

3. - Neste ponto é necessário fazer uma associação mental da função de cada tecla após a leitura do texto (Figura 4).

- Violação do ponto 6 “Consistência e Padrões” com gravidade 2.

1.2.4 Aquisição contínua:

Figura 5 - Sequências de menus para definir parâmetros do varrimento**.**

4. - Embora exista alguma prevenção de erros, como validação de valores e mensagens de erro acontece que após entrar num menu não é permitido cancelar, ou seja, não é permitido voltar atrás sem preencher os campos (Figura 5).

- Violação do ponto 5 "Prevenção de Erros" com gravidade 3.
- Violação do ponto 9 "Ajuda no reconhecimento, diagnóstico e recuperação de erros", com gravidade 3.

5. – Verificou-se junto do docente responsável pela utilização do equipamento que o menu "Aquisição descontínua" era desnecessário, visto que se obtém o mesmo resultado fazendo uma "Aquisição contínua", com a vantagem desta ser mais rápida.

- Violação do ponto 8. " Design estético e minimalista", com gravidade 2.

1.2.5 Gráficos

Ao seleccionar no menu inicial as teclas F1 ou F2 são exibidos os gráficos de radiação em coordenadas lineares (Figura 6) e polares (Figura 7) respectivamente. Permite ainda comutar entre eles ao pressionar a tecla do gráfico que se pretende visualizar.

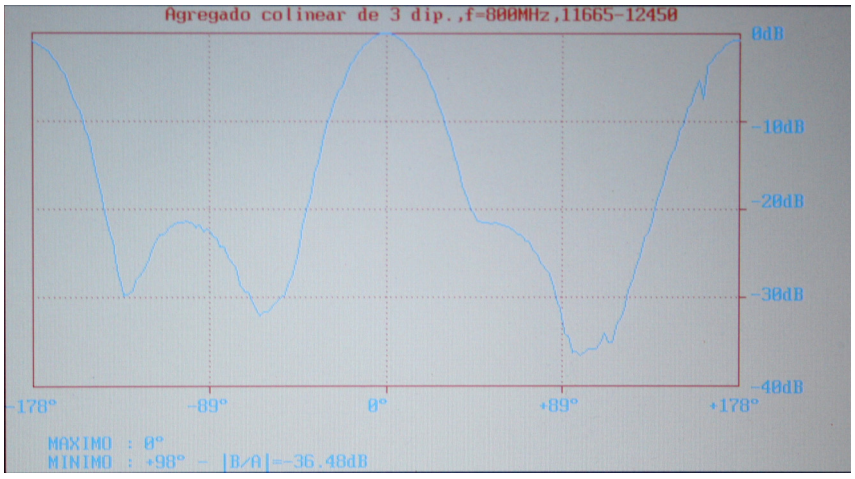


Figura 6 - Gráfico de radiação em coordenadas lineares.

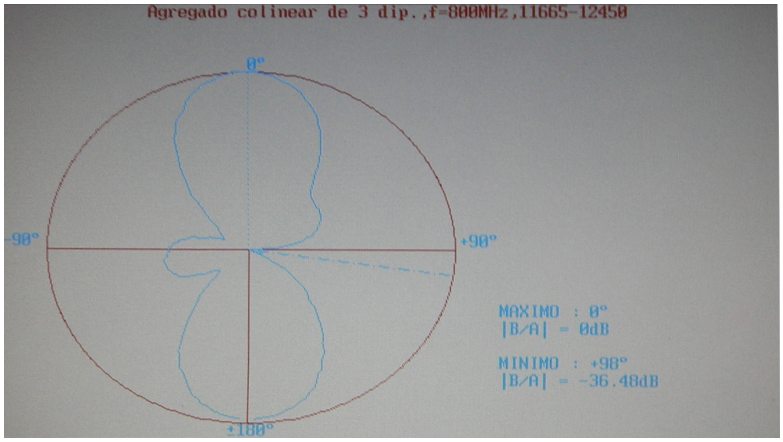


Figura 7 - Gráfico de radiação em coordenadas polares.

Podemos ainda mudar o título do gráfico, seleccionando a opção “Escolher mensagens para o gráfico” (M), no menu inicial (Figura 8).

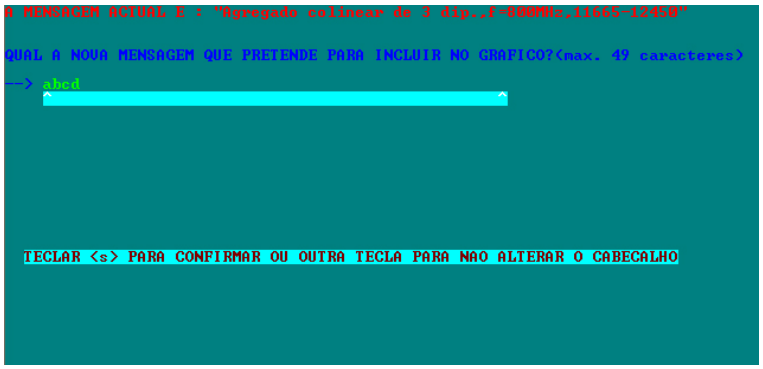


Figura 8 - Menu Escolher mensagens para o gráfico.

Assim, como podemos alterar a escala, na opção do menu inicial “Mudar a gama de valores do gráfico”, (tecla F5) (Figura 9).

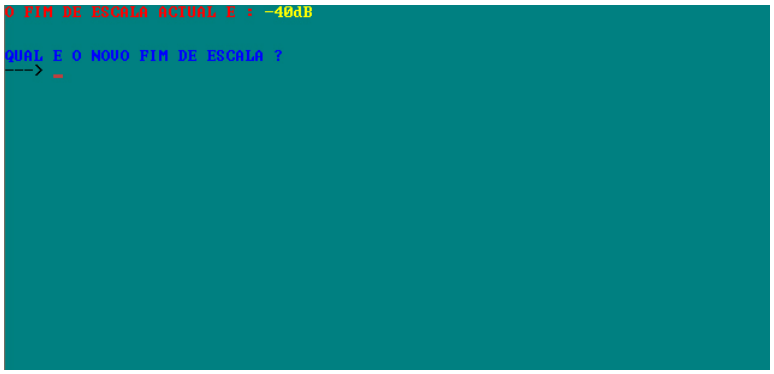


Figura 9 - Menu mudar a gama de valores do gráfico.

Não foi detectado nenhum erro novo e/ou grave em relação aos supra-mencionados.

1.2.6 Gravar e ler ficheiros.

Caso se pretenda gravar ou ler um ficheiro para ou de um directório que não o de *default*, temos de usar um menu dedicado apenas a essa função (Figura 10). O qual não permite visualizar os directórios existentes.

Ou seja, fazemos uso da memória, caso conheçamos a árvore de directórios existentes no disco a que pretendemos aceder, ou então usamos notas exteriores (método a que o docente utilizador habitual se via obrigado a recorrer).

Não é possível criar novos directórios.

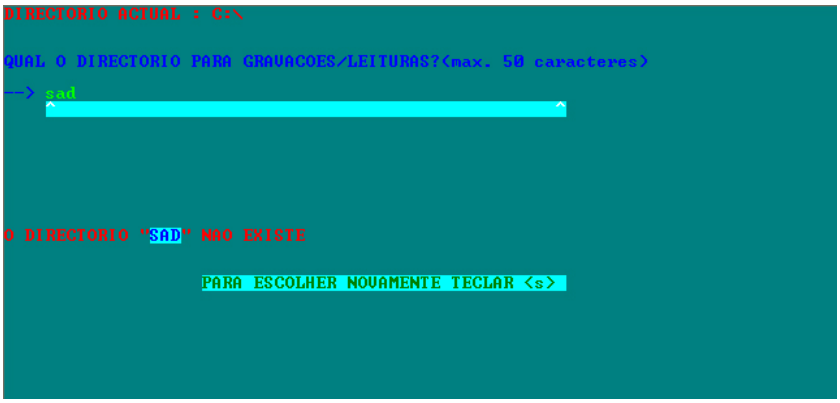


Figura 10 - Menu escolher directório para gravar/ler.

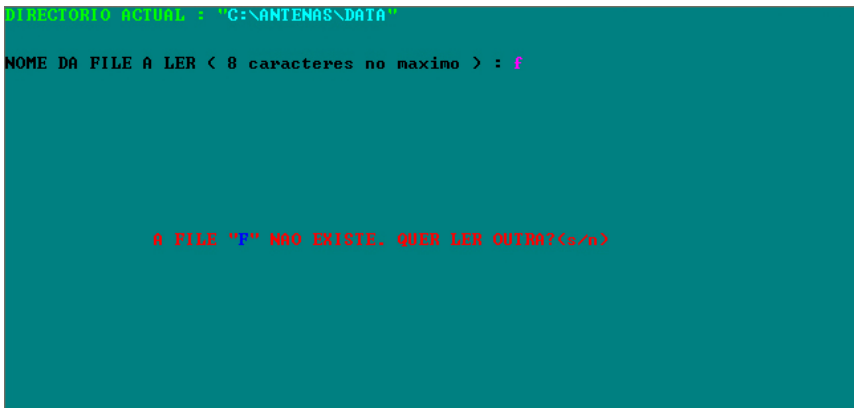


Figura 11 - Menu ler valores do disco.

6. - Processo de escolha de directório requer menu próprio. Requer a memorização da *path* exacta que se pretende, assim como o nome do ficheiro (Figura 11 e Figura 12).

- Violação do ponto 3 “Controlo e liberdade por parte do utilizador” e 6 “Prevenção de Erros”, ambos com gravidade 3.
- Violação do ponto 2 “correspondência entre o sistema e o mundo real”, com gravidade 2.

7. - Tamanho do nome de directorias limitado a oito caracteres (sistema DOS).

Violação do ponto 6 “Prevenção de Erros”, com gravidade 3.

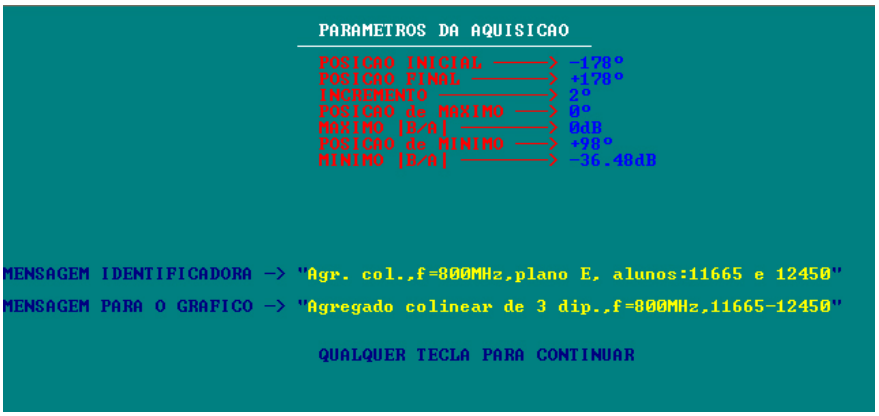


Figura 12 - Apresentação de dados do ficheiro lido a partir do disco.

1.2.7 Editar Valores das Medidas.

Existe a possibilidade de ver e editar um valor das medidas.

Os valores são exibidos tela a tela, percorridos com auxílio do teclado. Para editar um valor da tabela é necessário fornecer o ângulo e o novo valor (Figura 13).

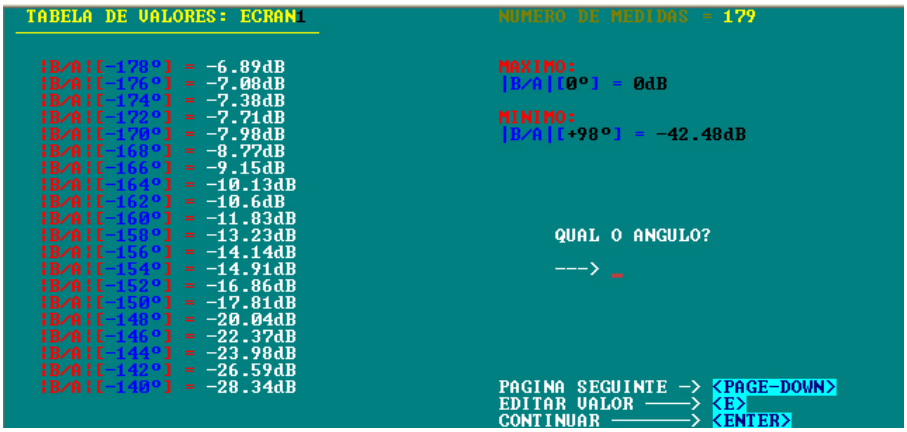


Figura 13 - Menu Editar/ver valores das medidas.

1.2.8 Apreciação Geral

De uma forma geral, as limitações detectadas prendem-se principalmente com o sistema operativo existente na altura, assim como com o *hardware*.

Não é sensível a maiúsculas e minúsculas, o que se considera uma facilidade.

Por exemplo, quando se pretende mudar de directório, a opção mais fácil é sair do programa, anotar os directórios e ficheiros que se pretende ler e voltar a entrar no sistema.

Verifica-se na maioria dos menus a existência de validação de valores de entrada pelo utilizador.

8. - Não existe no entanto validação de hardware, ou seja, se um dos dispositivos estiver desligado, o sistema reage com mensagens de erro que não são claras para um utilizador, como se pode ver a titulo de exemplo, na Figura 4, acabando por bloquear.

- Violação do ponto 5 “Prevenção de Erros” e 9 “Ajuda no reconhecimento, diagnóstico e recuperação de erros”, ambos com gravidade 3.

1.3 Estudo e concepção da Maqueta

1.3.1 Identificação de Objectivos de Usabilidade

O objectivo é criar um sistema de mais fácil utilização e mais intuitivo, para que abranja uma maior gama de utilizadores, incluindo aqueles com menores conhecimentos da área e do sistema em causa.

Além disso, pretende-se também corrigir todas as falhas de usabilidade detectadas no *software* anterior e oferecer novas facilidades.

1.3.2 Identificação de Funcionalidades a oferecer pelo sistema

As funcionalidades que o novo sistema terá de oferecer serão, na sua essência, as mesmas, mas de uma forma mais fácil e intuitiva de usar (Diagrama 1).

No entanto, tirando partido da experiência do utilizador frequente do sistema, serão suprimidos menus que revelaram não ter utilidade, como é o caso do menu de aquisição descontínua.

Para mais, adicionam-se outras facilidades como comprimento dos nomes dos ficheiros, visibilidades de ficheiros e directórios na sua manipulação (evitando recurso à memória do utilizador), o *feedback* do *hardware* e a possibilidade de gravar os gráficos como imagens (jpg).

Ilustração do funcionamento do sistema:

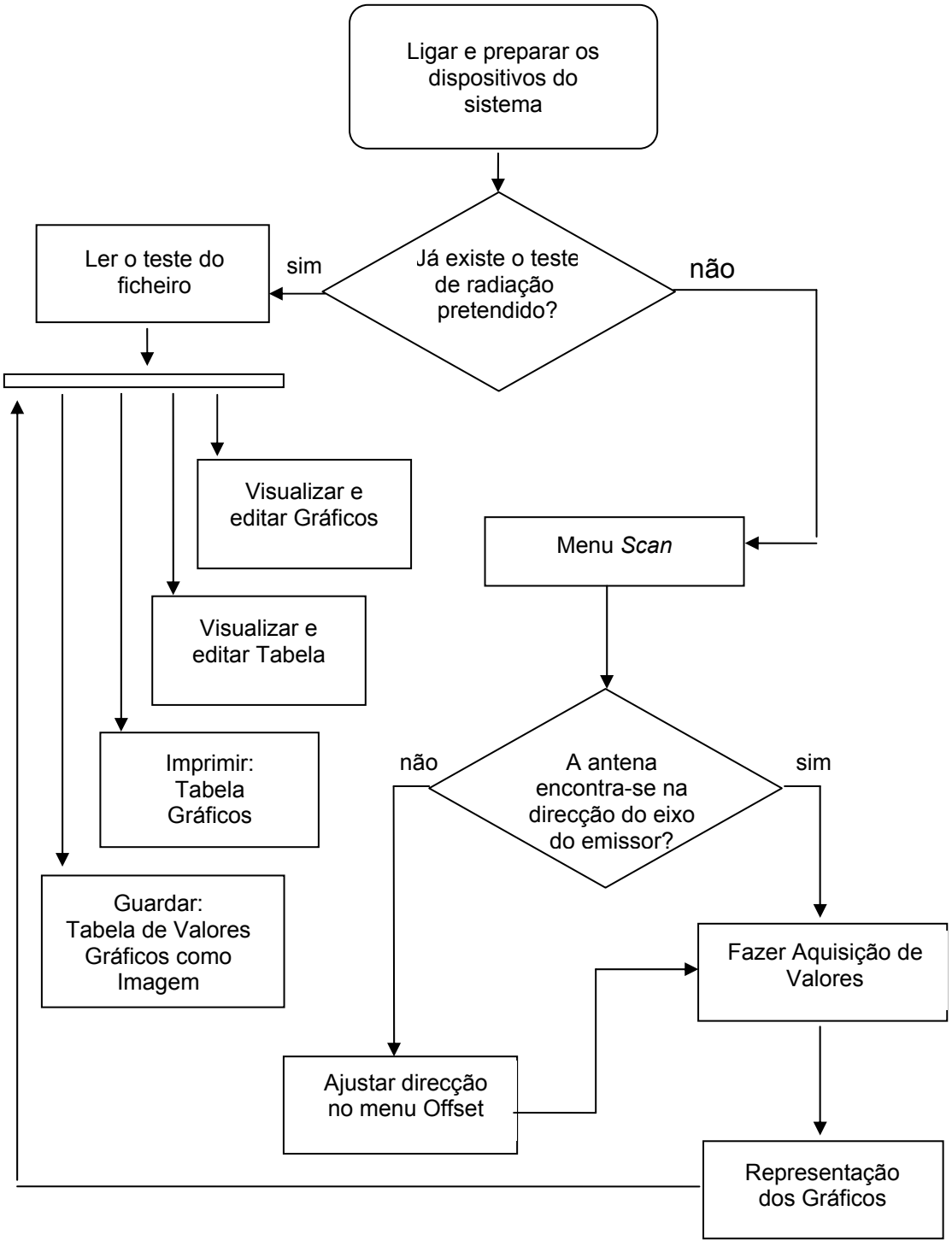


Diagrama 1 - Funcionalidades da maqueta RadScan.

1.3.3 Proposta de Modelo Conceptual

Propõe-se então a construção de um modelo conceptual do novo sistema. Este sistema embora não funcional apresentará aos utilizadores o "rosto" do novo *software* assim como simulará de forma aproximada as suas funcionalidades (Anexo E. - Código).

Considera-se esta a melhor forma de verificar se o novo *software* vai de encontro às expectativas dos utilizadores. Tem-se ainda a facilidade de colocar a maqueta à prova por utilizadores, para poder obter um *software* final bastante optimizado na óptica dos utilizadores.

De entre os sistemas operativos de ambiente de janelas de maior facilidade de utilização, temos o Windows e o Linux, no entanto hoje em dia a plataforma mais usada é o Windows. Seleccionando o Windows XP, estamos automaticamente a abranger maior número de utilizadores com à vontade neste ambiente, a aproveitar recursos existentes no laboratório e ainda a usar uma plataforma que apresenta mais estabilidade do que sistemas operativos mais recentes como o Windows Vista. Este facto só por si torna a aplicação mais fácil de utilizar, devido à forte familiaridade com o resto das aplicações usadas em ambiente Windows, bem como a possibilidade de uso do rato e de menus animados num ambiente agradável e personalizável.

O teclado continua usável graças à introdução de teclas de atalho nos menus. Verifica-se a importância de manter também uma familiaridade com o sistema já existente, optando-se então por criar os menus com os campos de introdução de valores distribuídos de forma semelhante, embora os menus tenham sido agrupados por categorias:

- Menu ficheiro, para gravar e ler ficheiros.
- Menu aquisição, onde podemos efectuar as tarefas relacionadas com as medições.

Correcção da orientação da antena com a antena emissora, introdução dos parâmetros das medidas e ainda a possibilidade de medir usando os parâmetros predefinidos.

- Menu visualização de medidas, que permite após a aquisição ter novo acesso aos gráficos e à tabela de valores com a possibilidade de edição dos mesmos.
- Adicionou-se um menu de ajuda que explica os procedimentos e menus necessários para executar as diferentes tarefas.

Após esta fase embrionária pretendemos numa fase posterior de desenvolver uma ajuda ao utilizador com um documento de auxilio: O Manual de Utilizador.

Foi introduzido ainda um barra de estados onde se pode ver em tempo real o estado do hardware: motor, voltímetro e o valor da correcção do eixo. Mais tarde será introduzida também a posição da antena ao longo da sua rotação.

O software foi desenvolvido com a ferramenta de C# do Visual Studio.Net da Microsoft (c). A este software foi decidido dar o nome de "RadScan" e daqui para a frente neste texto será assim que ele será referido, sendo a sua ilustração patente nas Figura 14 a Figura 19.

1.3.4 Plano de Avaliação

Após a elaboração da maqueta é necessário testá-la para se descobrir onde e como se pode melhorar o *software*, de forma a ser de fácil utilização, versátil e atractivo (*user friendly*).

Usar-se-á dois processos: a avaliação heurística e um teste de usabilidade.

1.3.4.1 Avaliação Heurística

O *software* foi desenvolvido de forma a obedecer aos “10 critérios de usabilidade” descritos por Jacob Nielsen [7].

De seguida proceder-se-á à análise dos aspectos a ter em conta:

- Visibilidade do estado do sistema

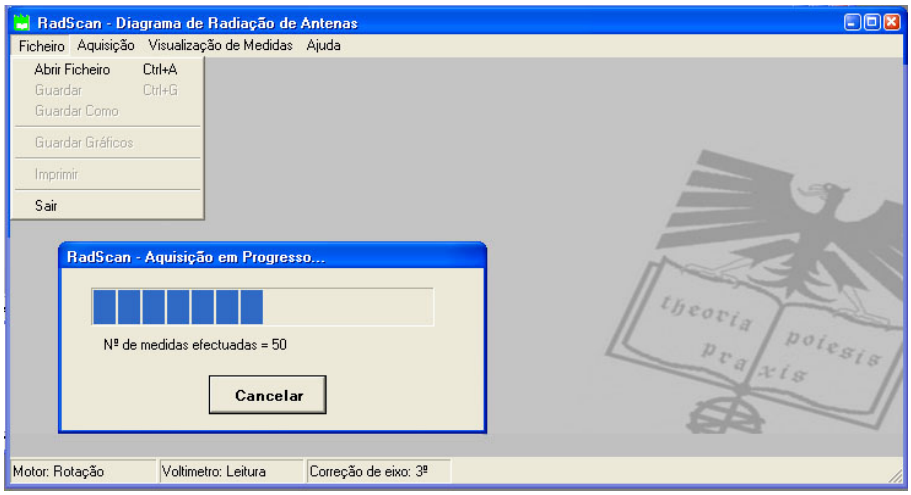


Figura 14 - RadScan, exemplos de visibilidade do sistema.

Existe uma barra de progresso que fornece o número de medidas efectuadas, a noção do volume de medidas realizadas e as ainda em falta (Figura 14).

A barra de tarefas fornece informação sobre o estado do hardware.

Menus inactivos transmitem directamente a informação da indisponibilidade das operações.

- Correspondência entre o sistema e o mundo real

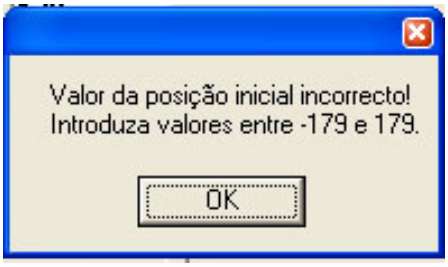


Figura 15 - RadScan, mensagem de erro.

Todos os menus utilizam o Português em linguagem de utilização corrente, as mensagens são claras e de fácil compreensão, mesmo para utilizadores sem qualquer experiência.

No entanto, existe apenas um botão de "OK" que é em inglês, um erro de gravidade 0, visto não apresentar qualquer dificuldade e a sua resolução é complexa (Figura 15).

- Controlo e liberdade por parte do utilizador

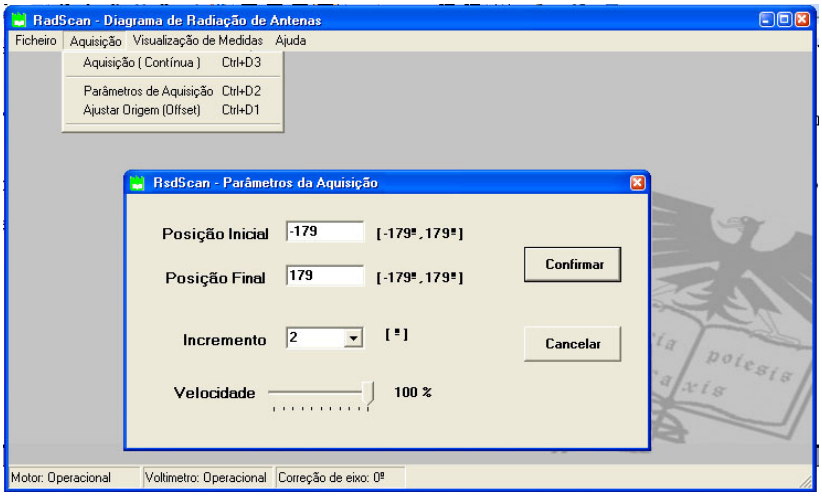


Figura 16 - RadSan, menu e janela de diálogo da aquisição.

A organização por menus e sub-menus permite ao utilizador escolher qualquer função sem ser submetido a uma sequência de procedimentos (Figura 16 na parte superior).

Em qualquer menu existe sempre a hipótese de cancelar e voltar ao menu anterior, ou à janela inicial do *software*.

Permitirá ainda fazer aquisição de dados, sem ter de configurar novamente os parâmetros em medições sucessivas ou usar parâmetros predefinidos, (Aquisição (Contínua) Figura 16)

- Consistência e Padrões

Como se pode verificar na janela de "Aquisição", patente na Figura 16 a configuração de parâmetros para a aquisição de dados mantém-se consistente com os existentes na versão anterior do *software* para MS-DOS.

- Prevenção de Erros

A prevenção de erros começa pelo facto de tornar inactivas funções não permitidas (ficando a funções em tom cinza claro) (Figura 14). Desactiva-se a possibilidade de introdução de caracteres em campos que requerem apenas algarismos, ao premir uma tecla de uma letra esta não aparece no campo, ver Figura 16.

Temos ainda mensagens de erro, exemplo na Figura 15, que variam conforme o erro cometido, para proporcionar uma boa percepção do erro cometido pelo utilizador.

Criou-se um filtro que impossibilita abrir ou gravar ficheiros com extensão que não a tratada pelo *software* (.xl).

- Reconhecer em vez de recordar

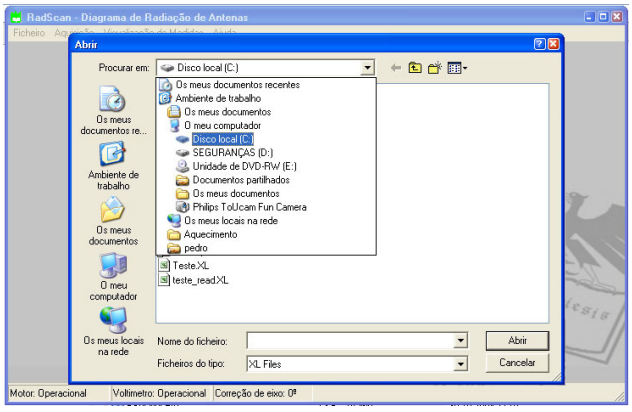


Figura 17 - RadScan, Janela de diálogo abrir ficheiro.

Quando pretendemos ler ou gravar um ficheiro, simplesmente basta procurar o ficheiro, ou o caminho (*path*) pretendido, respectivamente, isto em janelas do sistema operativo e familiares de outras aplicações do Windows.

Há a possibilidade de criar novas pastas, apagar ou manipulá-las.

Corrige-se assim neste ponto uma grave falha do sistema anterior.

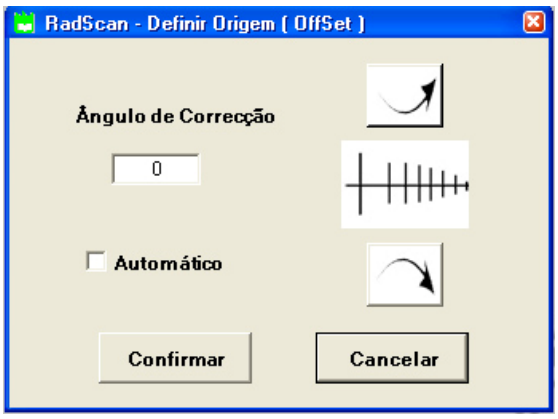


Figura 18- RadScan, Janela de correcção de eixo

Na janela de diálogo de correcção do eixo de orientação (para colocar a antena na posição central da rotação) (Figura 18), colocou-se nos botões setas com o sentido de orientação em vez de texto. Pretende-se que a percepção do sentido de rotação seja imediato, não deixando sequer espaço para confusão entre esquerda e direita.

- Uso flexível e eficaz

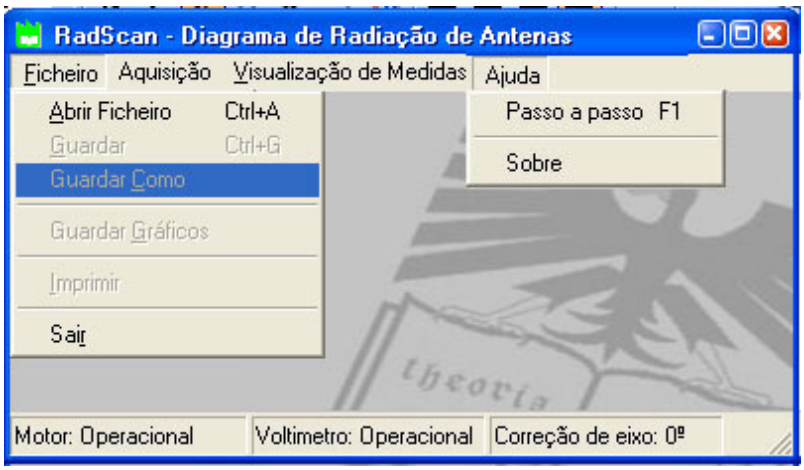


Figura 19 - RadScan, menu ficheiro e ajuda.

Há aceleradores para utilizadores experientes:

- Atalhos nos menus, estes ficam visíveis ao pressionar a tecla Alt, podendo ser actuados pressionado a letra sublinhada no respectivo menu.

- Atalhos rápidos, para as funções mais usadas Ctrl + (a chave do menu do antigo *software*) permite que utilizadores do antigo sistema consigam rapidamente usar os atalhos rápidos (Figura 19).

Existe ainda a possibilidade de usar as teclas “ENTER” e “ESC” nas janelas, como as funções dos botões de aceitar e cancelar, respectivamente. Existe a possibilidade de medições sem necessidade de repetir configurações.

- Design estético e minimalista

  Todas as janelas de diálogo contêm apenas a informação necessária para a tarefa em execução, não havendo portanto qualquer competição com outras informações.

- Ajuda no reconhecimento, diagnóstico e recuperação de erros

  Como foi referido num ponto anterior, as mensagens de erro são claras e diversificadas conforme o tipo de erro colocando o cursor no campo a corrigir para que seja fácil a recuperação do erro.

- Ajuda e Documentação

  Existe um menu de ajuda que descreve os passos a seguir para a tarefa a realizar. Na janela de ajuda temos as tarefas organizadas no manual do utilizador.

1.3.4.2 Teste de usabilidade.

Contou-se com a colaboração de alguns alunos e docentes do curso de Electrónica e Telecomunicações sem qualquer experiência na utilização de qualquer uma das versões do *software.* Para a realização do teste de usabilidade, foi-lhe solicitado que realizasse as tarefas a seguir descritas.

Tarefas:

1. Em ambiente DOS, aceda ao *software* com:

   cd medidas <Enter>
   cd med_vect <Enter>
   mede-BA <Enter>

   Em ambiente Windows use o atalho para "RadScan" existente na área de trabalho.

2. Fazer uma amostragem contínua de -175º a 175º, medidas com intervalo de 2º e uma velocidade de 75%

3. Visualizar resultados:
   a. Tabela de Valores
   b. Gráfico em coordenadas polares
   c. Gráfico em coordenadas lineares
4. Gravar amostragem com o nome: "23457957"

5. Sem consultar a página anterior, leia o ficheiro gravado na alínea anterior.

6. Leia outro ficheiro qualquer existente no disco em c:\

Observação:

Apresentam-se de seguida a título ilustrativo um dos resultados obtidos através da observação da execução das tarefas:

- **Ambiente MS-DOS:**

Tarefas:	Tempo Previsto	Tempo Realizado	Erros	Ajudas	Tarefa Cumprida
1	20s	50s	0	0	N
2	4m	3m	1	1	S
3	1m	1m40	0	0	S
4	30s	44s	0	0	N
5	2m	4m30	2	2	N
6	50s	46s	0	0	S

Tabela 1 - Resultados da execução das tarefas em ambiente DOS.

- **Ambiente Windows:**

Tarefas:	Tempo Previsto	Tempo Realizado	Erros	Ajudas	Tarefa Cumprida
1	20s	5s	0	0	S
2	4m	1m30	0	0	S
3	1m	30s	0	0	S
4	30s	10s	0	0	S
5	2m	12s	0	0	S
6	50s	6s	0	0	S

Tabela 2 - Resultados da execução das tarefas em ambiente Windows.

Tal como podemos observar pelos valores médios das tabelas a implementação do *software* em ambiente Windows evitou a ocorrência de erros e ajudas, bem como reduziu substancialmente os tempos necessários à execução de cada tarefa. No novo software, o tempo total de interacção do utilizador com a interface reduziu-se para 22% e todas as tarefas foram executadas com sucesso.

2 Plataforma de desenvolvimento

Neste capítulo, são analisados três aspectos fundamentais para o desenvolvimento do *software*, a saber: o sistema operativo, a linguagem de programação e o protocolo de comunicação entre os dispositivos.

No capítulo 1.3.3 seleccionou-se o Windows como sistema operativo por razões de usabilidade, especialmente por existir um maior universo de utilizadores de Windows, familiarizados com as suas aplicações.

Na verdade, existem outros sistemas operativos com ambiente gráfico, como é o caso do Linux, mas a maior vantagem seria o facto de se tratar de um *software* grátis. Nesta fase, por razões de implementação, a utilização do Windows, e mais concretamente do Windows XP, revela-se novamente mais vantajosa, devido à limitada oferta de *drivers* por parte do fabricante da placa GPIB instalada no computador destinado ao controlo. Um sistema operativo como o Windows Vista está limitado no sistema de medição em questão, por um lado, pelo PC, que não cumpre os requisitos mínimos deste SO e, por outro, pela placa de controlo de GPIB. Esta teria de ser substituída por uma placa mais recente. Além disso, deve optar-se nesta situação pelo SO mais estável e fiável, neste caso o Windows XP.

Após definir o SO a utilizar, é necessário primeiramente seleccionar uma linguagem de programação que seja proveitosa para o desenvolvimento do *software*; em seguida, analisar o funcionamento do protocolo de comunicação GPIB e, finalmente, observar compatibilidades entre ambos.

2.1 Selecção de Linguagem de Programação

Quanto à linguagem de alto nível em ambiente Windows, existem, entre outras, C++, Java, Visual Basic, C#, .Net, Delphi, Pascal, Fortan ou Prolog.
No vasto universo de linguagens de programação para ambiente Windows, procurou-se uma linguagem recente de forma a promover a portabilidade do código na escala temporal, facultando a compatibilidade com sistemas operativos futuros. Outro aspecto tido em conta nesta escolha foi a familiaridade com a linguagem C, dominante durante o percurso académico. Por estes motivos, optou-se pela linguagem C#.

C# é uma linguagem de alto nível, orientada a objectos, e faz parte da plataforma .Net desenvolvida pela Microsoft e tem na sua génese as linguagens C++ e Java. O seu funcionamento assemelha-se à linguagem de programação Java e a sua sintaxe e vocabulário ao C. A linguagem C# foi criada juntamente com a arquitectura .NET. Embora existam várias outras linguagens que suportam essa tecnologia (tais como VB.NET, C++, J#), C# é considerada a linguagem de referência do .NET, pois foi a primeira a ser desenvolvida, sendo criada quase inteiramente de raiz de modo a funcionar nesta plataforma. De facto, grande parte das classes do .NET Framework foi desenvolvida em C# [8].

2.2 Interligação e comunicação entre equipamentos

Quanto à interligação dos diferentes equipamentos, confirmou-se que a comunicação GPIB seria a melhor opção no sistema apresentado. Isto porque embora alguns dos equipamentos utilizados permitam também a comunicação série, o GPIB é mais rápido e é comum a todos os dispositivos, incluindo o computador destinado ao controlo, o qual está munido de uma placa ISA / GPIB.

Esta placa, por outro lado, apresenta uma limitação do sistema aos novos dispositivos GPIB com interface USB, visto que o *driver* mais actual que suporta estes dispositivos não suporta esta placa, AT-GPIB/TNT.

2.2.1 O protocolo GPIB

Em 1965, a Hewlett-Packard desenvolveu o sistema *Hewlett-Packard Interface Bus* (HP-IB) para interligar os seus equipamentos programáveis aos seus computadores. Surgiu da necessidade de dar resposta à complexa coordenação entre vários equipamentos. Devido a sua alta taxa de transferência (normalmente 1Mbytes/s), esta interface rapidamente ganhou popularidade. Foi reconhecida mais tarde como norma IEEE 488 (1975), evoluiu até à norma ANSI/IEEE 488.1 (1987). Actualmente designado por *General Purpose Interface Bus* (GPIB), tem uma utilização mais ampla que o HP-IB. O ANSI/IEEE 488.2 (1987) reforçou a norma original, definindo como os instrumentos e controladores comunicam. A especificação *Standard Commands for Programmable Instruments* (SCPI) adoptou a estrutura de comandos definida na norma IEEE 488.2, **[9]** criando um único e mais compreensível conjunto de comandos programáveis, usado em qualquer instrumento SCPI **[14],[15],[16].**

2.2.2 Tipo de mensagens GPIB

Os dispositivos GPIB comunicam entre si enviando mensagens de interface e mensagens específicas desse equipamento:

- As mensagens de interface controlam o barramento. São normalmente denominadas por mensagens de comando. Estas executam funções como a inicialização do barramento, endereçamento, definição de modos de programação remota ou local.

- As mensagens específicas do dispositivo, frequentemente designadas por mensagens de informação, contêm informação específica do dispositivo como instruções de programação, resultados de medidas, o estado da máquina, e ficheiros de informação.

- O termo comando usado anteriormente não deve ser confundido com algumas instruções de dispositivos que também são designados por comandos, são apenas mensagens de informação na interpretação da interface GPIB.

2.2.3 Controladores, equipamentos emissores e receptores

Os dispositivos GPIB podem ser emissores, receptores e/ou controladores, o emissor envia mensagens de informação a um ou mais receptores. O controlador gere o fluxo de informação no barramento, enviando comandos para todos os dispositivos. A título de exemplo podemos considerar um voltímetro digital, que funciona tanto como emissor como receptor.

O GPIB assemelha-se ao barramento de um computador com a diferença que o barramento neste, interliga placas através da placa mãe enquanto o GPIB interliga dispositivos autónomos através de cabos *standard*.

O objectivo do controlador GPIB é comparável ao de um CPU num computador, mas uma melhor analogia será comparar o controlador a uma central de comutação de um sistema telefónico, que monitoriza a rede de comunicações (barramento GPIB). Quando a central reconhece que alguém quer comunicar interliga o emissor com o receptor. O controlador endereça normalmente um receptor antes do emissor poder enviar uma mensagem ao receptor. Depois de a mensagem ser transmitida, o controlador pode endereçar outros emissores ou receptores.

Algumas configurações GPIB não requerem controlador. Por exemplo, um dispositivo que seja sempre emissor, designado por dispositivo *talk-only*, é ligado a um dispositivo *listen-only*, que só recebe.

O controlador é necessário quando o endereço do emissor ou receptor tiver que ser alterado. As funções do controlador são usualmente controladas através de um computador.

Um computador equipado com hardware e *software* de GPIB poderá desempenhar qualquer um dos papéis *Talker*/*Listener* ou Controlador.

Apesar de podermos ter vários controladores no sistema GPIB, apenas um poderá estar activo, chamado de *Controller-IN-Charge* (CIC). No entanto um controlador activo poderá comutar do actual CIC para um controlador inactivo. Apenas um controlador activo se pode nomear como CIC. Normalmente o controlador activo é a placa GPIB.[1]

[1] Encontram-se em anexo informações complementares sobre as características eléctricas físicas do sistema GPIB.

2.2.4 SCPI

Em Abril de 1990, um grupo de construtores de instrumentos anunciaram a especificação SCPI, que define um conjunto comum de instruções para programação de instrumentos. Antes da SCPI, cada fabricante de instrumentos desenvolvia seu próprio conjunto de comandos para os seus instrumentos. A vaga de normalização forçou os criadores de sistemas de testes a aprender diferentes conjuntos de comandos e parâmetros específicos de dispositivos para os vários instrumentos de vários fabricantes usados na aplicação, resultando em programações complexas, atrasos imprevistos e elevados custos. Definindo um conjunto de comandos comum aos vários fabricantes e dispositivos, a SCPI reduz o tempo de desenvolvimento e aumenta a legibilidade dos programas de teste assim como a facilidade de substituir instrumentos.

A norma SCPI é completa e ainda extensível, esta unifica os comandos de *software* de programação de instrumentos. A primeira versão da norma remonta a 1990, actualmente o consórcio SCPI continua com a adição de comandos e funcionalidades da norma SCPI. Esta, tem o seu próprio conjunto de comandos em adição aos comandos e consultas mandatários comuns do IEEE488.2.

Contudo a norma IEEE488.2 é usada como sua base, a SCPI define os comandos de programação que se podem usar com qualquer tipo de hardware ou dispositivo de comunicação.

A norma SCPI especifica regras para a abreviação das palavras de comando e utiliza o protocolo de troca de mensagens da norma IEEE488.2 assim como suas regras e formatos de comandos e parâmetros. Poderão ser usados comandos na sua forma longa (MEASure) ou na sua forma abreviada em letras maiúsculas (MEAS).

A utilização da norma SCPI oferece inúmeras vantagens aos engenheiros de teste. Uma das vantagens é ser um conjunto de comandos de fácil compreensão, interpretação de funções que abrangem a maior parte dos dispositivos de teste. O facto de existir um conjunto comum de comandos assegura um alto nível de compatibilidade entre instrumentos e consequente facilidade na substituição de equipamentos, assim como reduz o esforço no desenvolvimento de novos

sistemas de teste. O conjunto de comandos da SCPI é hierárquico, assim, é fácil a integração de comandos para funções mais específicas ou novas.

2.2.5 O modelo de instrumentos SCPI

Como resultado de compatibilizar e categorizar grupos de comandos, a norma SCPI define um modelo de instrumento programável, esse modelo está ilustrado no Diagrama 2 e é aplicado a todos os diferentes tipos de instrumentos.

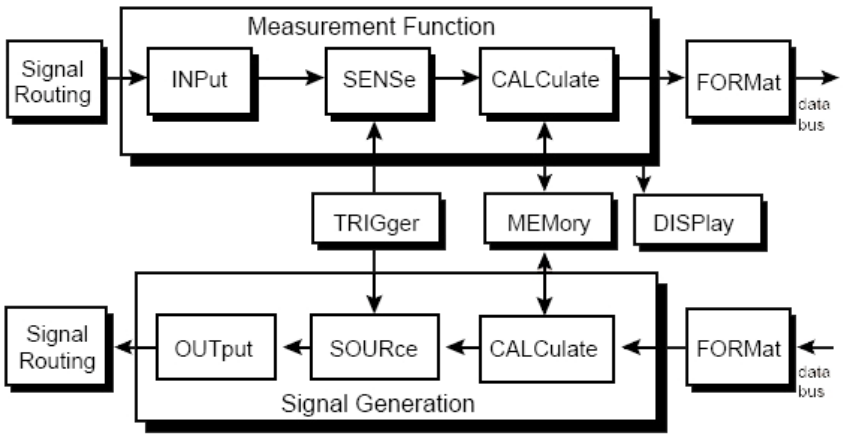


Diagrama 2 - Modelo de instrumento SCPI.

Nem todos as componentes de funcionalidades deste modelo são usadas necessariamente em todos os instrumentos. Por exemplo um osciloscópio não tem a funcionalidade definida pelo bloco de geração de sinal do modelo de instrumento SCPI.

A norma SCPI define a hierarquia entre conjunto de comandos para controlar uma funcionalidade específica dentro de cada um destes blocos funcionais.

A componente de ***signal routing*** controla a ligação do sinal às funcionalidades internas dos instrumentos; a componente de ***measurement*** transforma o sinal numa forma pré-processada (dados) e a componente de ***signal generation*** transforma dados internos dos instrumentos em sinal do mundo real. A componente de ***memory*** destina-se a guardar dados internamente nos instrumentos. A componente de ***format*** transforma a informação dos instrumentos no formato adequado a transmitir pelo barramento GPIB. A componente ***trigger***

sincroniza as acções dos instrumentos com funções internas, eventos externos e outros dispositivos.

A função de ***measurement*** permite o mais alto nível de compatibilidade entre dispositivos, devido a que esta função tem de obedecer às especificações dos parâmetros passados e não às funcionalidades internas dos instrumentos. Na maioria dos casos é possível substituir um determinado instrumento que realiza uma determinada medida por outro instrumento capaz de realizar a mesma medida sem que para isso seja necessário alterar o comando SCPI.

A componente de ***MEASurement*** está subdividida em três partes distintas: INPut, SENSe e CALCulate. O bloco de INPut condiciona o sinal recebido antes de ser convertido em informação pelo bloco de SENSe. O bloco de INPut inclui funções de filtragem, polarização (biasing) e atenuação. O bloco de SENSe converte o sinal em informação interna, passível de ser manipulada, perante parâmetros como: gama de valores, resolução, janela de tempo de aquisição e possibilidade de utilização de rejeição de modo comum. O bloco de CALCulate converte a informação adquirida no formato mais conveniente para uma dada aplicação, estas transformações são do tipo de: conversão de unidades, tempo de queda do sinal (*fall time)*, tempo de reposição de sinal a nível alto (*rise time)* e parâmetros de frequência.

A componente de geração de sinal, transforma informação em sinais físicos de saída. O modelo de instrumento SCPI divide a geração de sinal em três blocos funcionais: OUTPut, SOURce e CALCulate.

O bloco de OUTPut condiciona o sinal de saída após este ser sido gerado e inclui funções de filtragem, polarização e atenuação.

O bloco de SOURce gera o sinal em função de características específicas e de informação interna tais como: parâmetros do sinal, amplitude de modelação, potência, intensidade de corrente, tenção e frequência.

O bloco CALCulate converte informação da aplicação na forma favorável a correcta geração de sinal, configurando parâmetros como: correcção de efeitos externos, conversão de unidades e mudança de domínios.

2.2.6 Exemplo de um comando SCPI

Para ilustrar um comando SCPI, nada melhor que apresentar a sequência de comandos usados numa das configurações do instrumento de medida, o voltímetro de amplitude e fase, usado no nosso sistema de teste RadScan., visto usar esta tecnologia.

- *Strings* de configuração do voltímetro:

```
*RST
FORMat LOGarithmic
TRIGger:SOURce BUS
SENSe AVOLtage
SYSTem:FORMat ASCii
*TRG
```

- *Strings* de leitura de amplitude:

```
MEASure? AVOLtage
```

- *Strings* de leitura de fase:

```
MEASure? PHASE
```

3 Desenvolvimento do *software* de controlo

3.1 Rotinas de comunicação com equipamentos GPIB

Atribui-se a designação de funções de baixo nível às funções que tratam da comunicação e do controlo directo do *hardware* [12], neste caso da interacção com o *driver* da placa GPIB e com os restantes dispositivos externos. Para a sua elaboração foi necessário compreender a linguagem de cada um dos dispositivos: controlador e voltímetro. Ambas são baseadas em *strings* trocadas sobre o protocolo GPIB. No Diagrama 3 descreve-se os protocolos de comunicação utilizados com os diferentes dispositivos do sistema.

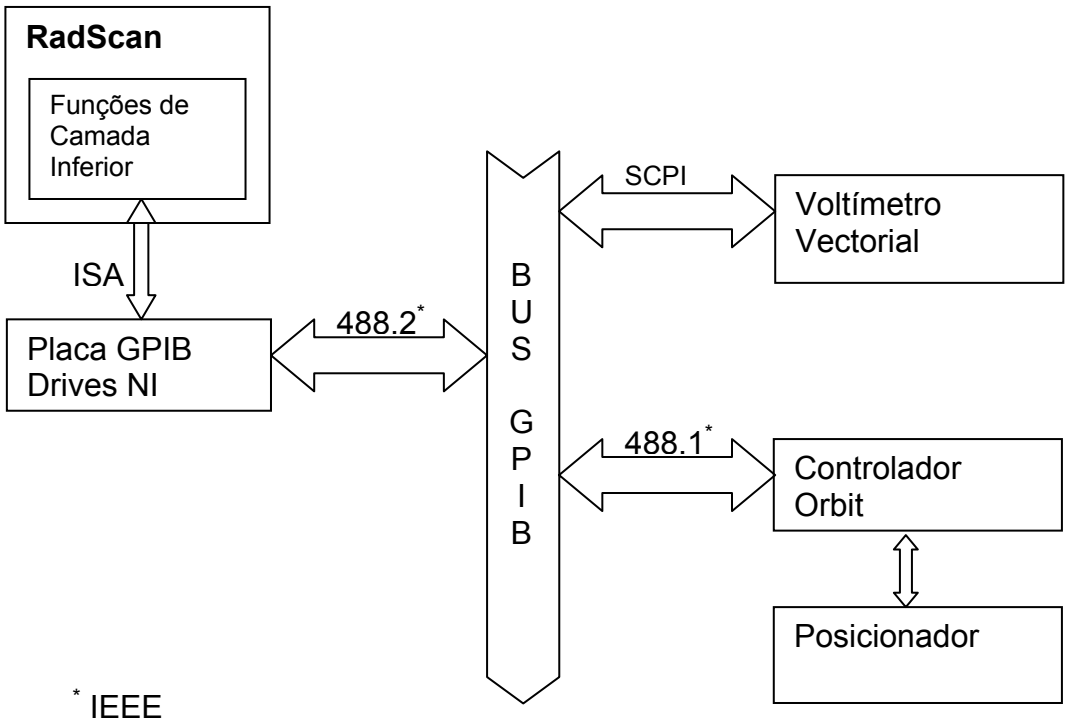


Diagrama 3– Ilustração dos protocolos de comunicação com os dispositivos

3.1.1 Ferramentas de Comunicação

Para se proceder a esta análise e testar a comunicação entre os equipamentos GPIB, foi fundamental a utilização de algumas ferramentas. A primeira foi a "*Max*" ou "*National Instruments Measurement & Automation Explorer*" (Figura 20), *software* do *driver* da placa GPIB.

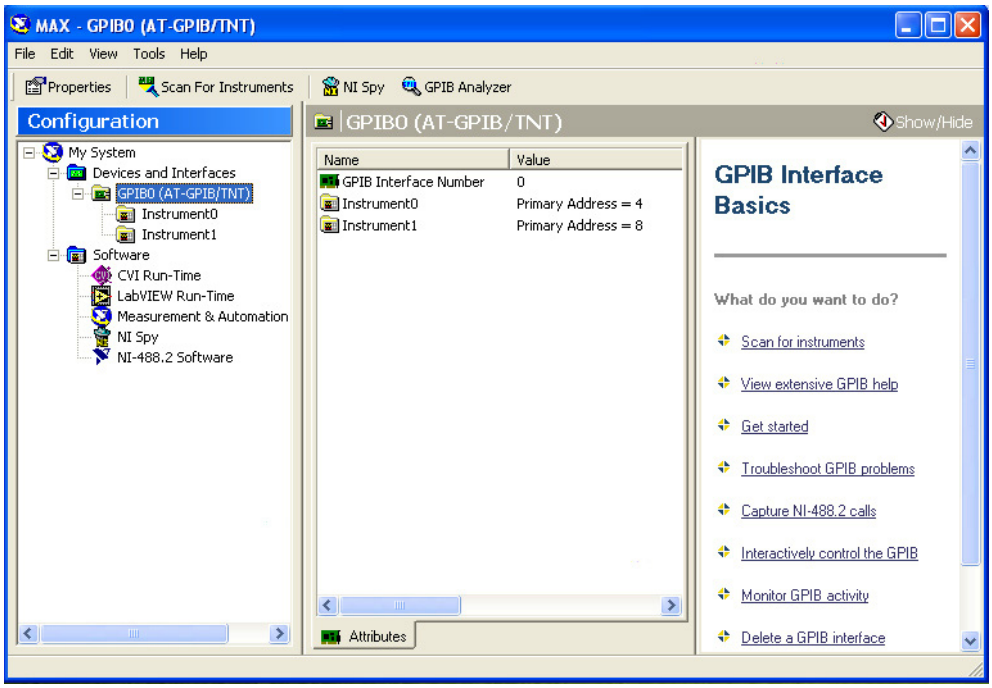


Figura 20 - National Instruments Measurement & Automation Explorer.

Este *software* disponibiliza duas valiosas aplicações:

- *"Communicator"* (Figura 21) , usado para enviar e receber comandos isolados para o bus GPIB;

- "*NI Spy*" (Figura 22), usado para "espiar" todas as s*trings* e os seus formatos, correspondentes aos comandos enviados para o *bus* GPIB e recebidos dos dispositivos, bem como as funções dos *drivers* usadas para tal e ainda eventuais erros e os seus tipos.

Outra ferramenta importante na clarificação da comunicação GPIB, desta vez com o controlador do posicionador, foi o "*Commlink*" (Figura 23), *software* de ambiente

Windows fornecido pelo fabricante do equipamento, a "Orbit Avanced Technologies".

3.1.2 Comunicação com o Voltímetro

Comunicar com o Voltímetro relevou-se fácil, visto trabalhar com a norma IEEE488.2. Para testar a comunicação com o voltímetro vectorial, usou-se o "Comunicator" do *software* da National Instruments. Ilustra-se, a título de exemplo, o envio do comando de pedido de identificação (`"*IDN?"`), ao qual o equipamento respondeu prontamente com o ID: `HEWLETT-PACKARD,8508A-050,0,2944` (Figura 21):

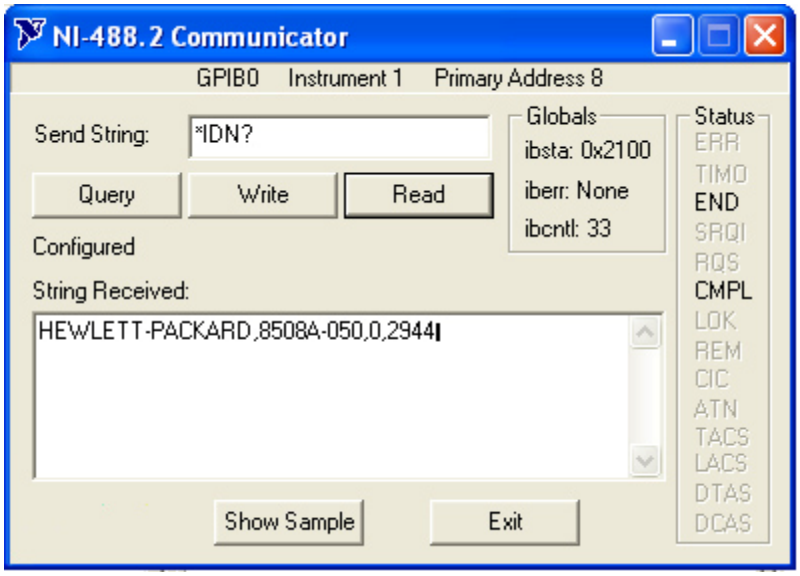


Figura 21 - Janela de comunicação isolada com bus GPIB.

Os comandos necessários para a aquisição de valores de medidas deste dispositivo são poucos e simples. O voltímetro é colocado em modo "*HP-IB triggering*", que faz leituras contínuas e fornece o valor da medida sempre que solicitada. Este está configurado para fazer leituras logarítmicas, responder a um *trigger* e dar a resposta em formato ASCII, para o *bus* GPIB. A selecção do modo de leitura A ou B|A é realizada também na configuração e estará disponível num dos menus do RadScan.

- **Caso da leitura do Canal A:**

  Configuração do Voltímetro

```
"*RST\nFORMat LOGarithmic\nTRIGger:SOURce BUS\nSENSe
AVOLtage\nSYSTem:FORMat ASCii\n*TRG"
```

```
Comando de aquisição de valor de medida:

"MEASure? AVOLtage"
```

- **Caso da leitura de Canal B em relação ao A:**

  Configuração do Voltímetro

```
"*RST\nFORMat LOGarithmic\nTRIGger:SOURce BUS\nSENSe BA\nSYSTem:FORMat
ASCii\n*TRG"
```

```
Comando de aquisição de valor de medida:

MEASure? BA
```

As rotinas de controlo dos dispositivos de aquisição de dados são baseadas em funções existentes na biblioteca do *driver* da placa GPIB. As *strings* de comando são inseridas em *arrays* para facilmente se proceder a alterações de configuração, tal como o modo da leitura ou outro dispositivo de medida. Desta forma será apenas necessário passar o índice do *array* correspondente ao modo a seleccionar ou a outro dispositivo de medida.

3.1.3 Comunicação com o Posicionador

A comunicação com este dispositivo relevou-se bastante complexa. Funciona no standard mais básico do protocolo IEEE488, os comandos enviados são específicos do fabricante e do modelo do controlador. O feedback e *status* do equipamento são obtidos através de leituras de bytes nos registos internos do

controlador. O fabricante não tem *drivers* deste equipamento para ambiente Windows, factor que obriga a criar rotinas que são de baixo nível.

Entenda-se que este sistema de posicionamento é composto por duas partes: o posicionador e o seu controlador. O posicionador é uma estrutura composta por motor e sensores. Esta estrutura move a antena nela acoplada. O controlador do dispositivo encontra-se no exterior da câmara que processa o controlo do posicionador e permite a utilização pelo utilizador em modo manual ou através do sistema GPIB[11].[2]

O teste de comunicação com o controlador iniciou-se de forma semelhante ao do voltímetro, isto é, enviando comandos (*strings*) compilados de acordo com a informação do manual do equipamento[2], usando para tal a mesma ferramenta, o *"Communicator"* (Figura 21). Apesar de não serem esperadas respostas do controlador (visto pertencer à norma IEEE488.1), esperava-se contudo um movimento concordante com as instruções. Tal não aconteceu. Dada esta situação, utilizou-se outra ferramenta do "*National Instruments Measurement & Automation Explorer*": o "NI spy". O "NI spy", foi usado para registar toda a informação que circulava através da placa GPIB, do qual se apresenta um trecho na Figura 22.

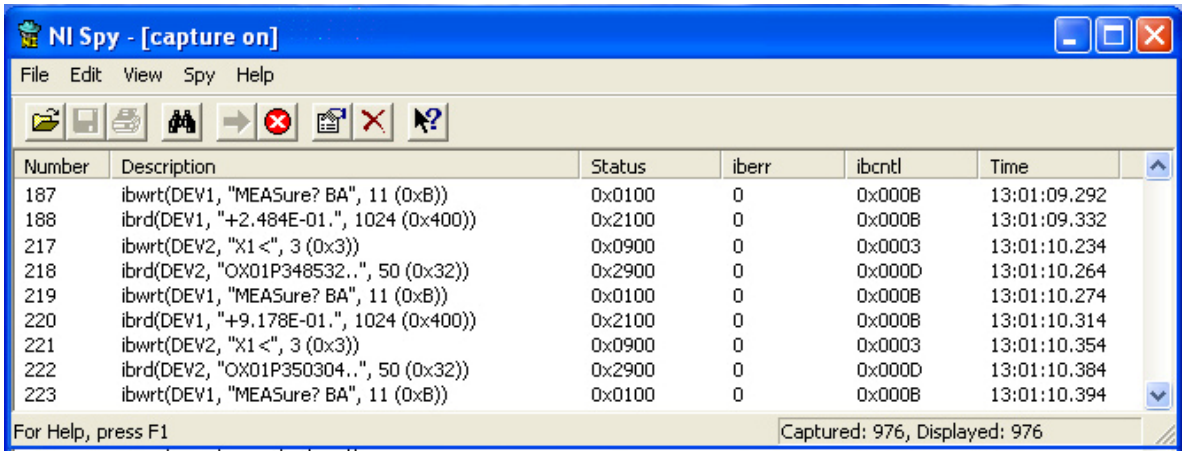


Figura 22 - NI Spy, captura.

[2] Nesta dissertação, considera-se este sistema como um todo, visto que a comunicação é efectuada apenas com a peça de controlo. Muitas vezes, por uma questão de facilidade de leitura dentro de determinados contextos, usa-se o termo de posicionador *lato sensu*, ou seja, inclui controlador e posicionador propriamente dito.

Para enviar informação, utilizou-se o *"Commlink",* uma ferramenta de controlo de movimentos. Pôde verificar-se que o controlador respondia unicamente ao primeiro comando e todos os posteriores resultavam em erro de *timeout*. Através da análise das instruções em hexadecimal, conclui-se que em alguns comandos seria necessário acrescentar à *string* de instrução dois caracteres invisíveis: o "CR" (0h0A) e o "Null" (0h00). No código do *software* desenvolvido em MS-DOS, apenas era acrescentado o carácter "Nul".

Depois de se compreender a estrutura correcta das *strings* de controlo, o desenvolvimento das rotinas de baixo nível para a comunicação com o posicionador foi elaborado usando o RadScan com análise através do "*NI Spy*".

3.1.3.1 Sintaxe de comandos do controlador Orbit

Tendo como referência principal o manual do posicionador e usando o "NI spy" para monitorizar a forma como o *software* "Commlink" funciona nas diferentes operações, consegue perceber-se de forma bastante precisa a maneira como se utilizam as *strings* dos comandos. As *strings* não têm formato standard, alterando a sua terminação consoante o modo de operação do controlador.

A forma de operar do posicionador é baseada em quatro modos: "Load", "Read", "Run" e "StandBy". No modo "Load", faz-se o carregamento de parâmetros, armazenando-os em memória interna; no modo "Read", procede-se à leitura de valores de parâmetros e estados internos do controlador; o modo "Run" concretiza o movimento do posicionador que foi programado na sua memória; o modo "Stand By" corresponde a um modo neutro no qual se encontra o posicionador quando é ligado ou quando se sai de qualquer um dos outros modos.

Destes quatro modos fazem parte vários comandos, dos quais se destacam os necessários para a realização do controlo do posicionador, agrupados nesta dissertação da seguinte forma: parâmetros de navegação entre modos de operação, parâmetros de movimento, leitura de registo de *status* e outros. Estes comandos são compostos por letras, siglas e os valores dos

respectivos parâmetros, os quais se vão agrupar de forma a criar as *strings* de mensagens de operação.

É de salientar a necessidade de, nos comandos de escrita, enviar as *strings* para o controlador com a terminação "\r\n". Na linguagem C, "\r" e "\n" correspondem ao caracter ASCII "CR" e "Nul", respectivamente. Para configurar os parâmetros de movimento em modo "**Load**", as *strings* deverão ter o caracter "<" imediatamente antes da terminação "\r\n".

Apresentam-se, de seguida, os comandos e as suas funcionalidades:

Parâmetros de navegação entre modos de operação:

L – Comando para entrar em modo "**Load"**;

S – Comando para entrar em modo "**Read"**;

G – Comando para entrar em modo "**Run"**;

H – Comando para sair de qualquer um dos modos anteriores para o modo de "**Stand By**".

Parâmetros de movimento:

Aa1 – selecciona o eixo a usar (neste caso, será sempre o 1);

Pat, Pes, Peb, Pet – têm de ser seguidos de 5 algarismos, sendo que o conjunto define a posição angular inicial;

Va00 – tem de ser seguido de 3 algarismos, sendo que o conjunto define a velocidade em percentagem;

Daf, Dar, Dad, Def, Der, Ded – seleccionam a sentido do movimento entre CW, CCW e a distância mais curta;

MR, MN, MT, MZ, UM – seleccionam o tipo de movimento entre **Raster Scan A, Raster Scan B, Track, Slew** e **Direct**, respectivamente.[3]

[3] Esta *string* é a única neste grupo em que a terminação não é composta pelo caracter "<", apenas "\r\n".

Leitura de registo de *status*:[4]

X1 - devolve uma *string* com o valor da posição, com o seguinte formato:

OX01Pzzzzzz, em que z corresponde ao valor da posição angular e a sua forma decimal será: zzz.zzz

X3 - devolve uma *string* com o valor da velocidade, com o seguinte formato:

OX01VyyyyyDX0Wkkkkk, em que z corresponde ao valor da velocidade angular e a sua forma decimal será: yyy.yy

X4 - devolve um byte (32bits) com o estado do controlador (em forma de *flag*), este byte é "mascarado" de forma a isolar as *flags* (bites) com a informação pretendida, sendo que se assinalam a vermelho aqueles que são usados no controlo pelo RadScan.

Os bits a verde são as principais flags responsáveis pelo controlo da versão em MS-DOS. Como podemos verificar, não coincidem com o controlo realizado pelo RadScan que controla os limites realizando leitura directa da posição angular.

X4 Registo de *Status,* 32 bits

STB1						B12	B11	B10	Eixos activos
							Axis		
STB2					B23	B22	B21	B20	Estados Gerais
					S/H	CPU	Limite CW	Limite CCW	
STB3	B37	**B36**			B33		**B31**	**B30**	Estados Fundamentais
	LOAD	RUN			READ		PC	STBY	
STB4	**B47**	**B46**	B45	**B44**	B43	B42	B41		Tipos de Movimentos
	Track	Raster A	Slew	Raster B	Slew CCW	Slew CW			

Tabela 3 - Descrição dos bits relevantes do byte de resposta ao comando X4.

[4] Os comandos que a seguir se descrevem - X1, X3 e X4 - poder-se-ão usar durante o movimento (modo "**Run**") com *strings* com terminação "<\r\n".

Outros parâmetros usados:

O – deve ser seguido de 5 algarismos, sendo que o conjunto define uma nova posição "zero" no controlador em relação ao seu zero físico (*Offset*);

Fe, Fz, Fb – activam e desactivam informação enviada para o *bus* GPIB durante o movimento do posicionador [2].

3.1.3.2 Reconfiguração dos parâmetros da memória não volátil do controlador Orbit

Para o correcto funcionamento do posicionador, a memória não volátil deve estar configurada de acordo com os parâmetros fornecidos pelo fabricante. Indicam-se em seguida os passos do procedimento a seguir para configurar estes parâmetros de memória:

- **Iniciar o CommLink** (Figura 23)

Figura 23 - Ícone do Commlink.

- **Ir ao menu "Configure" Comando "Axes" (**Figura 24**):**

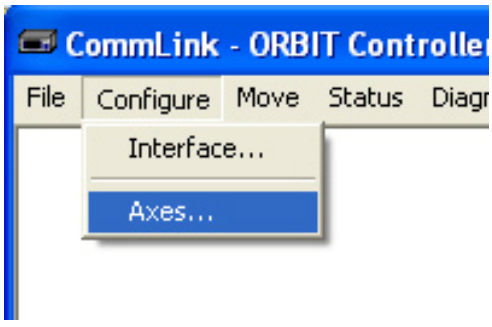


Figura 24 - Menu "File", comando "Axes"

- **Preencher os campos dos parâmetros com os valores adequados**

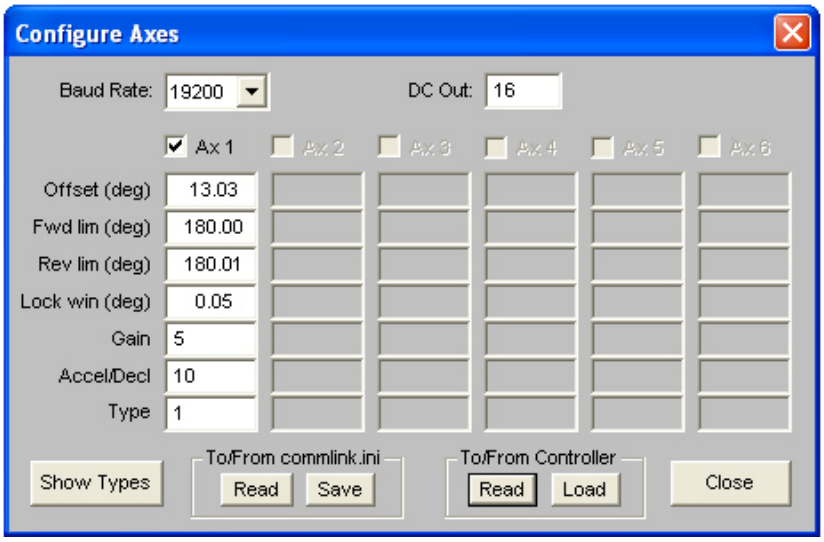


Figura 25 - Janela de configuração de parâmetros do controlador.

Os valores correctos dos parâmetros de memória do controlador são os seguintes (Figura 25)**:**

Offset(º) : (valor indiferente)[5] - define a posição "zero" do posicionador em relação ao seu zero físico.

FWd lim (º): 180.00 – tratam-se de valores-limite de rotação usados pelo controlador no cálculo do percurso mais próximo de forma a não atingir estes limites.

REV lim (º): 180.01 - *idem*[6]

Lock Win(º): 0,05 - *Lock Window*: 0º a 0,99º, intervalo em volta do qual o controlador assume ter atingido uma dada posição de paragem.

Gain: 5 - é o valor do ganho em malha fechada do controlo do movimento do motor.

Accel/Desc: 10 - é o valor da aceleração/desaceleração que define o factor de amortecimento em malha fechada do controlo do movimento do motor.

Tipo: 1 - define o tipo de tacómetro instalado no posicionador. Neste caso: *Dual speed synchro 0-360º*.

[5] Este valor é indiferente dado que será novamente alterado no início do RadScan.
[6] Se estes dois limites forem iguais (FWd lim e REV lim) serão desactivados.

- **Gravar alterações na memória não volátil** (Figura 26)

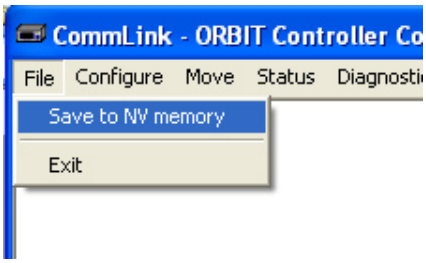


Figura 26 - Menu "File", comando "Save to NV memory".

3.1.3.3 Tipos de movimento possíveis

O posicionador possibilita vários tipos de movimento. Por uma questão de rigor e de metodologia, optou-se por analisar todos os movimentos disponíveis, inclusive aqueles usados na versão do *software* em MS-DOS. Desta forma, pôde verificar-se quais os movimentos mais adequados a implementar no RadScan.

Recorreu-se novamente à ferramenta CommLink para controlar o posicionador e assim proceder a uma análise das suas características de programação e da reacção física do posicionador.

O teste de movimento está disponível no CommLink, no menu "Move" (Figura 27).

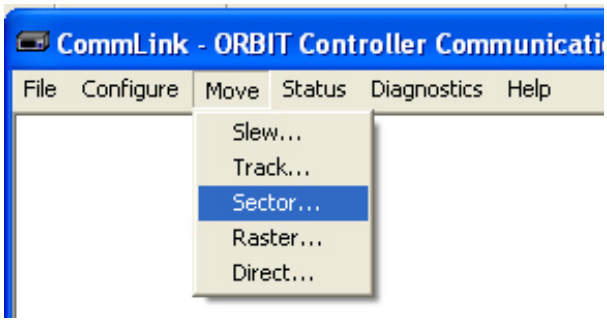


Figura 27 - Menu "Move" e seus comandos.

Os parâmetros dos vários tipos de movimento estão ilustrados pela janela correspondente no CommLink que permite executar tal movimento. Assim, pode encontrar-se diversos tipos de movimento. A saber:

- **Movimento Directo** [11][7] (Figura 28)

  Aplicação: controlo de sistemas dinâmicos com uso de feedback.

  Parâmetros de entrada:

  Eixo;

  Posição final;

  Velocidade.

  Acção: desloca o motor à velocidade definida para uma posição determinada, desacelerando quando se aproxima desta, sem no entanto parar efectivamente o movimento (movimento contínuo).

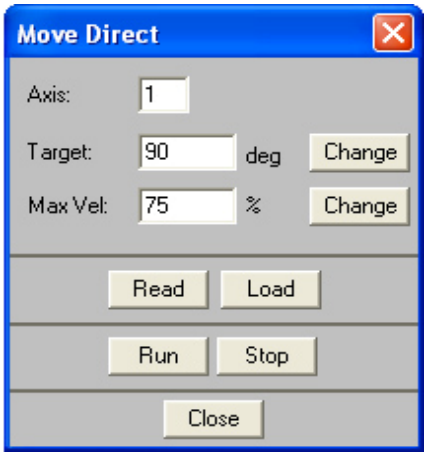


Figura 28 - Movimento "Direct", exemplo de configuração.

- **Track** [11][8] (Figura 29)

  Aplicação: movimentar para a posição determinada.

  Parâmetros de entrada:

  Eixo;

  Posição final;

  Velocidade;

  Sentido.

  Acção: desloca o motor para uma dada posição e pára o movimento quando atinge a posição-alvo.

[7] v. p.40.
[8] v. p.35.

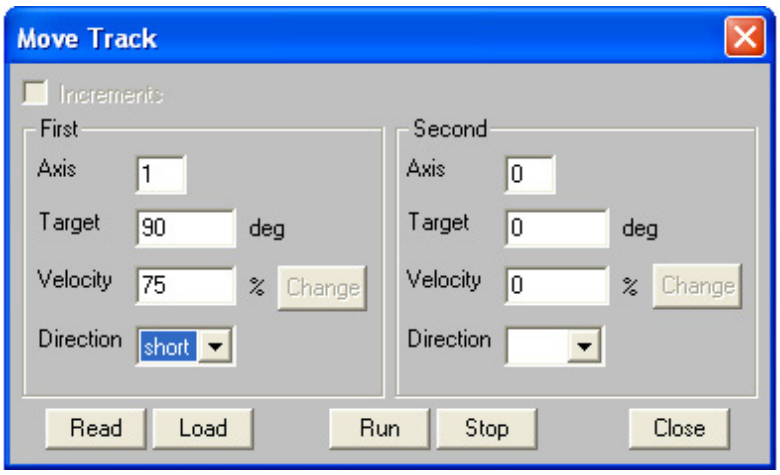


Figura 29- Movimento "Track", exemplo de configuração.

- **Sector: Raster A ou Raster B** [11][9] (Figura 30)

  Aplicação: movimento adequado à aquisição de medidas, realizado a velocidade constante.

  Parâmetros de entrada:

  Eixo;

  Posição Inicial;

  Posição final;

  Velocidade;

  Sentido;

  Incremento;

  Rampa de aceleração/desaceleração;

  Modo de incremento.

  Acção: movimenta o motor da posição inicial para a final, com velocidade constante. Tem necessidade de espaço angular para aceleração e desaceleração. Quando activa a opção, envia informação para o *bus* GPIB sempre que passa numa posição correspondente a um incremento.

[9] v. p.36.

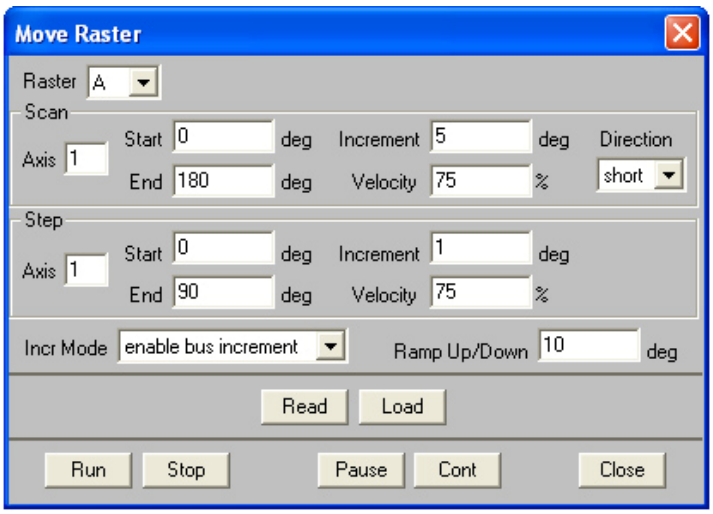


Figura 30 - Movimento "Raster", exemplo de configuração.

- **Move Sector** [11] [10] (Figura 31)

  Aplicação: movimento adequado à aquisição de medidas e realizado a velocidade constante, permitindo a repetição de varrimentos

  Parâmetros de entrada:

  Eixo;

  Posição inicial;

  Posição final;

  Velocidade de *scan* (movimento pretendido);

  Velocidade auxiliar (movimento de posicionamento);

  Sentido;

  Incremento;

  Rampa de aceleração/desaceleração;

  Modo de Incremento;

  Número de *scans*;

  Tipo de sector, A ou B.

  Acção: movimenta o motor da posição inicial para a final com velocidade constante. Tem necessidade de espaço angular para aceleração e desaceleração. Quando activa a opção envia informação para o *bus* GPIB sempre que passa numa posição correspondente a um incremento.

[10] v. p.35-38.

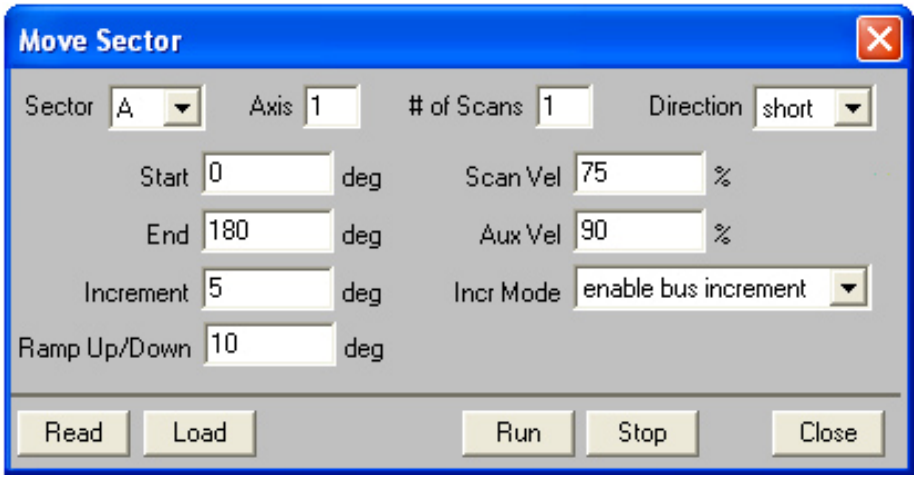


Figura 31 - Movimento "Sector", exemplo de configuração.

- **Move Slew** [11] [11] (Figura 32)

Aplicação: apresenta um movimento de rotação contínuo, permitindo configurar o sentido de rotação.

Parâmetros de entrada:

Eixo;

Velocidade auxiliar;

Sentido;

Acção: Coloca o posicionador em rotação com a velocidade e sentido definidos.

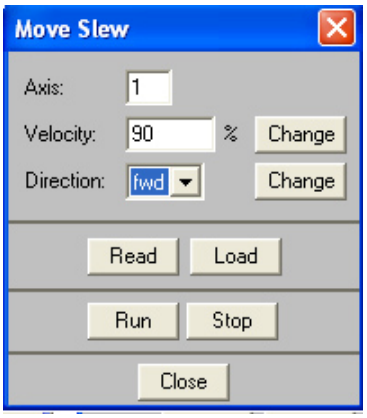


Figura 32 - Movimento "Slew" ,exemplo de configuração

[11] v. p.38.

3.1.3.4 Análise e escolha do movimento

No RadScan existem três fases que usam tipos de movimento diferentes: uma corresponde ao movimento de posicionamento da antena na posição inicial; outra ao acerto da posição central (zero); e uma outra corresponde ao movimento de aquisição de dados.

No movimento de posicionamento da antena na posição inicial de rotação, seleccionou-se o tipo de movimento "Move Direct" à semelhança do antigo sistema [4]. Nesta etapa, pretende-se unicamente direccionar a antena para o ângulo de início da rotação com rapidez. O "Move Direct" revelou-se, neste sentido, a escolha mais adequada, pois, devido às características já mencionadas, é unicamente necessário programar três parâmetros (o ângulo-alvo, o eixo e a velocidade). Nestas deslocações, usa-se a velocidade máxima e o sentido de rotação é definido pelo controlador do posicionador que efectua o percurso mais curto.

Numa outra etapa, quando se trata do acerto da posição central do movimento (*Offset*), já é necessário fornecer a direcção em que se pretende rodar a antena. Optou-se aqui pelo movimento "Move Sector", visto que neste caso é possível, para além dos parâmetros característicos do movimento precedente, definir o sentido de rotação. No RadScan incluiu-se um algoritmo que ajusta a velocidade de rotação consoante a amplitude de movimento que se pretende realizar, usando velocidades mais reduzidas para deslocações curtas.

Numa terceira etapa, durante a aquisição, é usado o movimento "Move Sector", que demonstrou ser amplamente versátil e responder às várias necessidades de aquisição. Visto que garante o movimento da antena a velocidade constante em toda a gama angular definida para a aquisição de dados. Neste processo implica que o posicionador inicie a marcha antes o primeiro ponto a medir criando espaço para acelerar e alcançar a velocidade definida.

3.2 Funções de comunicação com os dispositivos

Nas próximas páginas, descrevem-se somente as funções da camada inferior, responsáveis pela comunicação GPIB. Esta opção prende-se com o facto destas funções serem blocos-chave na execução do *software*.[12] No desenvolvimento de todo o código foi necessária a consulta de muita informação sobre as linguagens envolvidas, a qual se realizou maioritariamente a partir da Internet [5],[6],[8],[10].

Apresentam-se em seguida os cabeçalhos das funções da camada inferior, os seus parâmetros e respectivas descrições de funcionamento:

```
//*****************************************//
//*  Methods of GBIP Controller & Devices  *//
//*****************************************//

//*    Initialization of GPIB Controller //
public int HW_GPIB_BoradINI()

Variáveis de entrada: ---
Variáveis de Saída  : sucesso 0,erro -1
Descrição: Inicializa controladora GPIB.


//*********************************//
//*  Methods of Hardware Voltmeter  *//
//*********************************//

//** Initialition Voltmeter     **//
public int HW_VoltmeterInit(int modo)

Variáveis de entrada: modo de leitura do voltímetro A->0; A|B->1
Variáveis de Saída  : sucesso 0,erro -1
Descrição: Inicializa Voltímetro.


//**  Read the indentification of voltimiter **//
public string HW_VoltmeterID()

Variáveis de entrada: ---
Variáveis de Saída  : sucesso string com ID ,erro -1
Descrição: Lê identificação do dispositivo.
```

[12] Todo o código-fonte do *software* pode ser consultado no suporte digital anexo a esta dissertação.

```
//** Read Voltmeter value **//
public double HW_VoltmeterRead(int modo)
```

Variáveis de entrada: *index* do *array* com as configurações do modo ou dispositivo de leitura.
Variáveis de Saída : sucesso valor da medida, erro -100
Descrição: Lê do dispositivo.

```
// Take the board offline... //
public void HW_VoltmeterOff()
```

Variáveis de entrada: ---
Variáveis de Saída : ---
Descrição: Coloca dispositivo *Off-Line.*

```
//*******************************************//
//*  Methods of Hardware Position Controller  *//
//*******************************************//
///***  Try Hardware  ***///
/// Try Position Controller ///
public int HW_ControllerInit(double posicao_ini)
```

Variáveis de entrada: posição inicial (zero a defendido para o a arranque do sistema)
Variáveis de Saída : sucesso 0 ,erro -1
Descrição: Inicializa o controlador e coloca o posicionador na posição posicao_ini.

```
//* Controller Position *//
public double HW_ControllerPosition()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso valor da medida, erro -1
Descrição: Lê a posição angular do controlador Orbit.

```
//* Controller Velocity *//
public double HW_ControllerVelocity()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso valor da medida, erro 0
Descrição: Lê a velocidade angular do controlador Orbit.

```
//* Controler Word status *//
public int HW_ControllerStatus()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso stb3, erro -1
Descrição: Lê o byte stb3 da palavra de estado do controlador.

```
//*/ Direct Movement /*//
public void HW_ControllerDirect_Load(string position, string velocidade)
```

Variáveis de entrada: posição final, velocidade
Variáveis de Saída : ---
Descrição: Carrega parâmetros de movimento no controlador Orbit.

```
//* Sector Movement *//
public void HW_ControllerSector_Load(double start, double end, double
increment, double v_scan, int mode)
```

Variáveis de entrada: posição inicial, posição final, velocidade, incremento, e sentido.
Variáveis de Saída : ---
Descrição: Carrega parâmetros de movimento no controlador Orbit.

```
//* Track Movement - "Dad" Percurso + perto!*//
public void HW_ControllerTrack_Load(double end, double v_scan, string
direction)
```

Variáveis de entrada: posição final, velocidade e sentido.
Variáveis de Saída : ---
Descrição: Carrega parâmetros de movimento no controlador Orbit.

```
/*// Star Controller Movement //*/
public void HW_ControllerGo()
```

Variáveis de entrada: ---
Variáveis de Saída : ---
Descrição: Inicia o movimento do posicionador do controlador Orbit.

```
//*/ Stop Controller /*//
public int HW_ControllerStop()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso 0, erro -1
Descrição: Para (em qualquer parte ou tipo de movimento) o movimento do posicionador do controlador Orbit.

```
/// Turn Off Controller ///
public int HW_ControllerOff()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso 0, erro -1
Descrição: Coloca controlador Orbit *Off-Line*.

```
// Define o zero na posição passada (offset) //
public void offset_set(double offset)
```

Variáveis de entrada: ---
Variáveis de Saída : ---
Descrição: Define um novo zero (posição central) controlador Orbit.

```
//* Controler Word status *//
public int  HW_ControllerStatusX4()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso stb2, erro -1
Descrição: Lê o byte stb3 da palavra de estado do controlador Orbit.

```
//* Controler Word status *//
public bool HW_ControllerStatus_Standby()
```

Variáveis de entrada: ---
Variáveis de Saída : *stand by* verdadeiro, restante falso
Descrição: Verifica se o controlador Orbit está no estado de *stand by*.

```
//* Controler Word status array *//
public int [] HW_ControllerStatusArray()
```

Variáveis de entrada: ---
Variáveis de Saída : sucesso *array* de estado, erro *array* todo a 0
Descrição: Constrói nova palavra de estados num *array* de inteiros com as *flags* dos *bites* de interesse.

```
// Converts a double to a 5 char string
// the last 2 chars stand for the decimal part
private string d32_to_strig_orbit(double B)
```

Variáveis de entrada: valor decimal
Variáveis de Saída : *string* no formato requerido pelo controlador Orbit.

Descrição: Converte um valor decimal numa *string* com o formato exigido nos comandos do controlador Orbit (*string* de 5 caracteres numéricos em que os 2 caracteres de menor peso correspondem à parte decimal, não existe vírgula ou ponto).

3.3 Desenvolvimento de comando e interface

3.3.1 Aquisição de dados e interface com o utilizador

A principal função do RadScan é a aquisição de dados. Esta função é crítica, pois é necessário que ocorra em tempo real. Por conseguinte, os recursos de processamento não devem competir com outras aplicações. É por esta razão que no controlo se usa um computador pessoal com o mínimo de software e aplicações necessárias (ver o capítulo 5.1.2 Requisitos).

A aquisição de dados na versão em MS-DOS era realizada através de interrupções, as quais permitiam a aquisição de medidas em tempo real. Programava-se o controlador para enviar uma mensagem para a placa GPIB no PC sempre que passasse nos pontos do trajecto da rotação onde se completava o valor do incremento definido. O software reagia com uma leitura de amplitude, obtendo assim uma medida em cada ponto de incremento.

Num sistema baseado em Windows, as interrupções não são apenas geradas pelo *hardware*, mas há também interrupções desencadeadas pelo próprio sistema operativo e aplicações existentes. Ora estes dois tipos de interrupções vão competir entre si. O resultado será então a existência de listas extensas de pedidos de interrupção com tamanhos variáveis. Fica-se assim numa situação de incerteza em relação ao tempo de atendimento do pedido. Desta forma, o uso de interrupções revela-se pouco pertinente, pois o Windows não garante resposta às aplicações em tempo real. Existe ainda incompatibilidade entre algumas rotinas de interrupção da livraria do *driver* e o Windows XP, porque se trata de um *driver* desenvolvido para versões anteriores do Windows.

Neste contexto, e por tudo o que foi exposto, construiu-se a nova rotina de aquisição de dados usando o método de *polling*.

No início da aquisição, configura-se o voltímetro vectorial para uma leitura contínua de valores de amplitude e de fase. Por sua vez, o RadScan executa,

entre a posição inicial e final, um ciclo de leitura contínua de valores angulares do posicionador da antena. Quando detecta a passagem por pontos que correspondem ao incremento que define a cadência das medidas, o RadScan regista os valores capturados pelo voltímetro vectorial.[13]

Quer a aquisição de dados, quer as restantes funções do RadScan (manipulação do sistema, edição de tabela de dados, leitura ou gravação de ficheiros, entre outras) são tarefas que se realizam percorrendo as barras e menus do *software* de uma forma em tudo semelhante a qualquer aplicação de ambiente Windows.

Nesta altura, tem-se já uma ideia de como funciona o *software* em desenvolvimento pela leitura do Capitulo 1.3.4.1 Avaliação Heurística. Como resultado desta avaliação, juntamente com a Avaliação de Usabilidade e os pareceres dos utilizadores, a interface do RadScan é, nesta fase, alvo de consideráveis melhorias em relação à maqueta inicial, melhorias que abaixo se descrevem.

- Os menus "Imprimir" e "Gravar Gráficos" são suprimidos, visto que os dados adquiridos são destinados a pós-processamento.
- O menu de "Parâmetros de aquisição" é fundido num único menu "Aquisição", no qual surge sempre a janela para introdução dos parâmetros já com os últimos valores utilizados pré-preenchidos.[14]
- O processo de aquisição é ordenado nesta mesma janela, ao seleccionar o botão "Confimar".
- O menu intitulado "Ajustar Origem (*Offset*)" é substituído por uma designação mais breve, "Origem (*Offset*)". Neste menu, as imagens nos botões de direcção não se mostraram tão intuitivas quanto se esperava, razão pela qual são substituídas pelas palavras "esquerda" e "direita".
- Um campo com o valor actual de amplitude é introduzido também no menu "Origem (*Offset*)", de forma a facilitar o alinhamento da antena com o valor máximo da amplitude de radiação.
- A facilidade de exportar a tabela de dados obtidos directamente para o Excel foi acrescentada. Assim, o menu "Exportar Excel" inicia o Excel e

[13] O processo de leitura de uma medida apresenta um tempo de execução médio de 50ms.
[14] Aquando da primeira utilização serão apresentados valores *default* pré-preenchidos.

exporta de forma automática os dados, recriando a tabela na folha de Excel e construindo o seu gráfico em coordenadas lineares. Este processo poupa alguns passos de exportação de dados criando logo a partida uma folha Excel para eventual pós-processamento, onde se poderão proceder a algumas análises simples e/ou personalização da tabela e gráfico com as ferramentas disponibilizadas pelo Excel. Esta função leva à supressão do menu "Gráfico", visto a função de visualização do gráfico não ser manipulável como é na folha de Excel.

- No menu "Dispositivos" foi acrescentada a possibilidade de ver o estado de cada dispositivo do sistema e reiniciá-los, levando-os ao estado inicial, de forma a eliminar alguns erros existentes. Permite-nos assim verificar se o equipamento está ligado, se o posicionador está no modo PC ou em manual ou detectar a ausência de qualquer sinal de RF.

Para obter o aspecto final desejado, e dado que os gráficos em coordenadas polares não são totalmente personalizáveis em Excel, optou-se por utilizar neste trabalho, unicamente coordenadas rectangulares.[15] De facto, a visualização gráfica dos dados, tem apenas um papel de verificação rápida da figura obtida em relação àquela que se esperaria obter.

[15] No âmbito deste projecto, não se considerou prioritário desenvolver as coordenadas polares. Seria interessante vê-las desenvolvidas posteriormente num outro contexto.

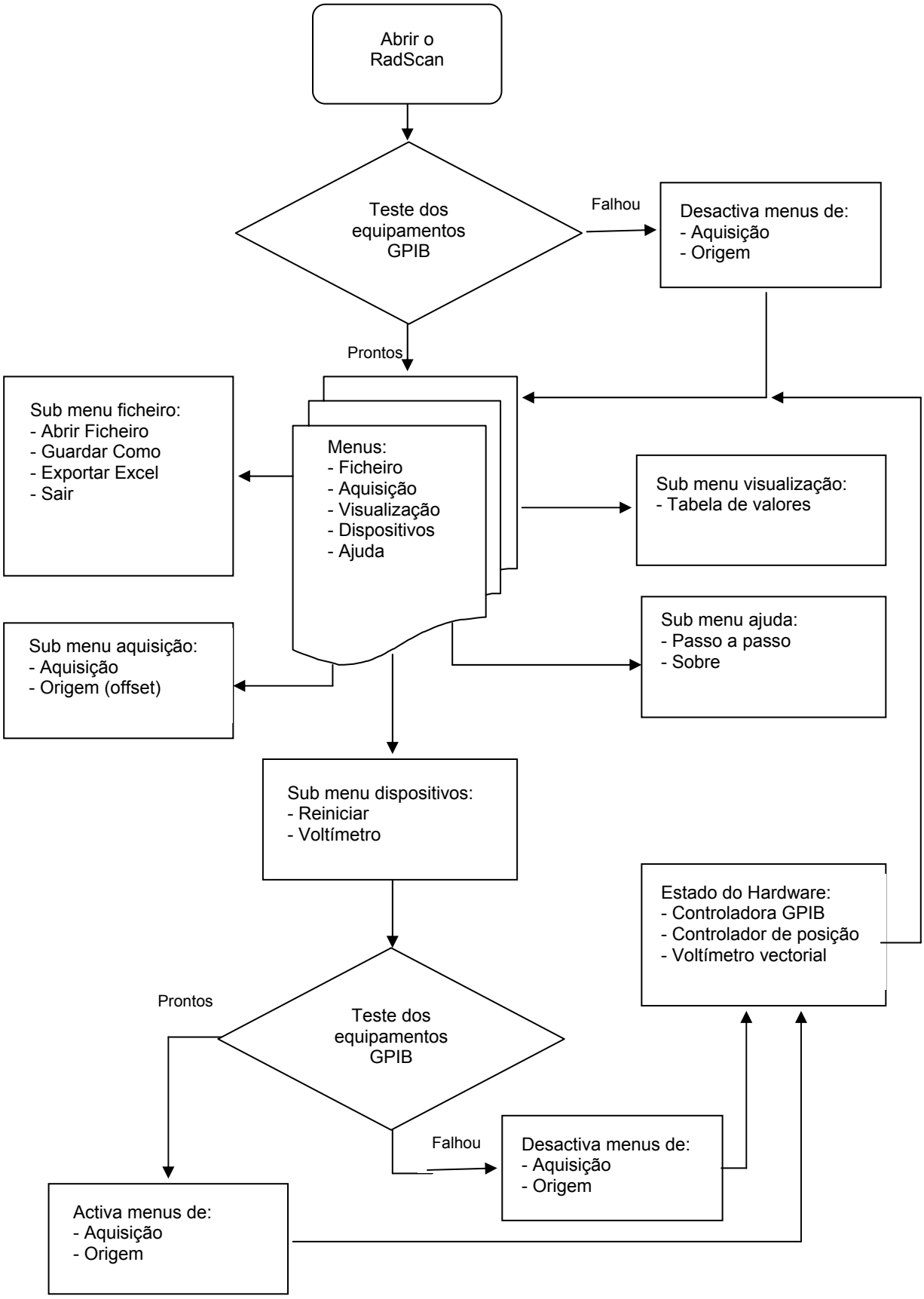


Diagrama 4 - Interacção entre menus e sub-menus no RadScan.

Todos os procedimentos de obtenção de tabelas e diagramas, ilustrações e especificações e explicação dos menus estão descritos em pormenor no Manual de Utilizador (em anexo). A disposição de menus e sub-menus está ilustrada no Diagrama 4.

A título de exemplo, na Figura 33 abaixo apresentam-se algumas janelas da interface do RadScan.

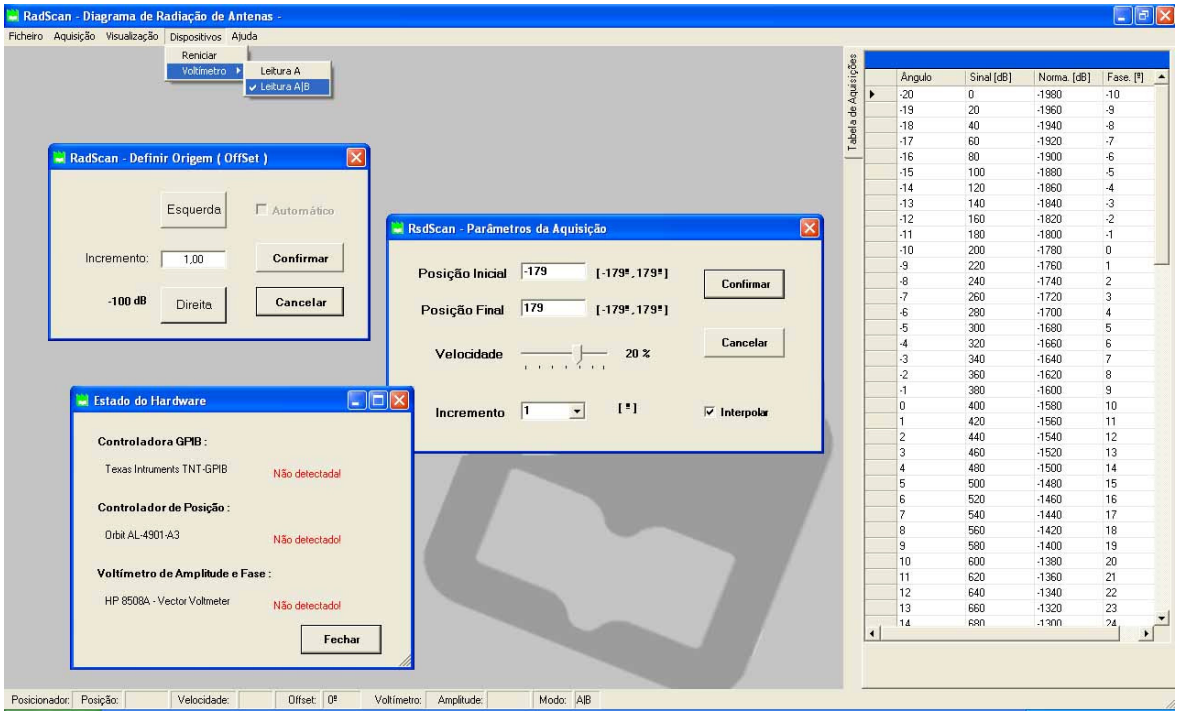
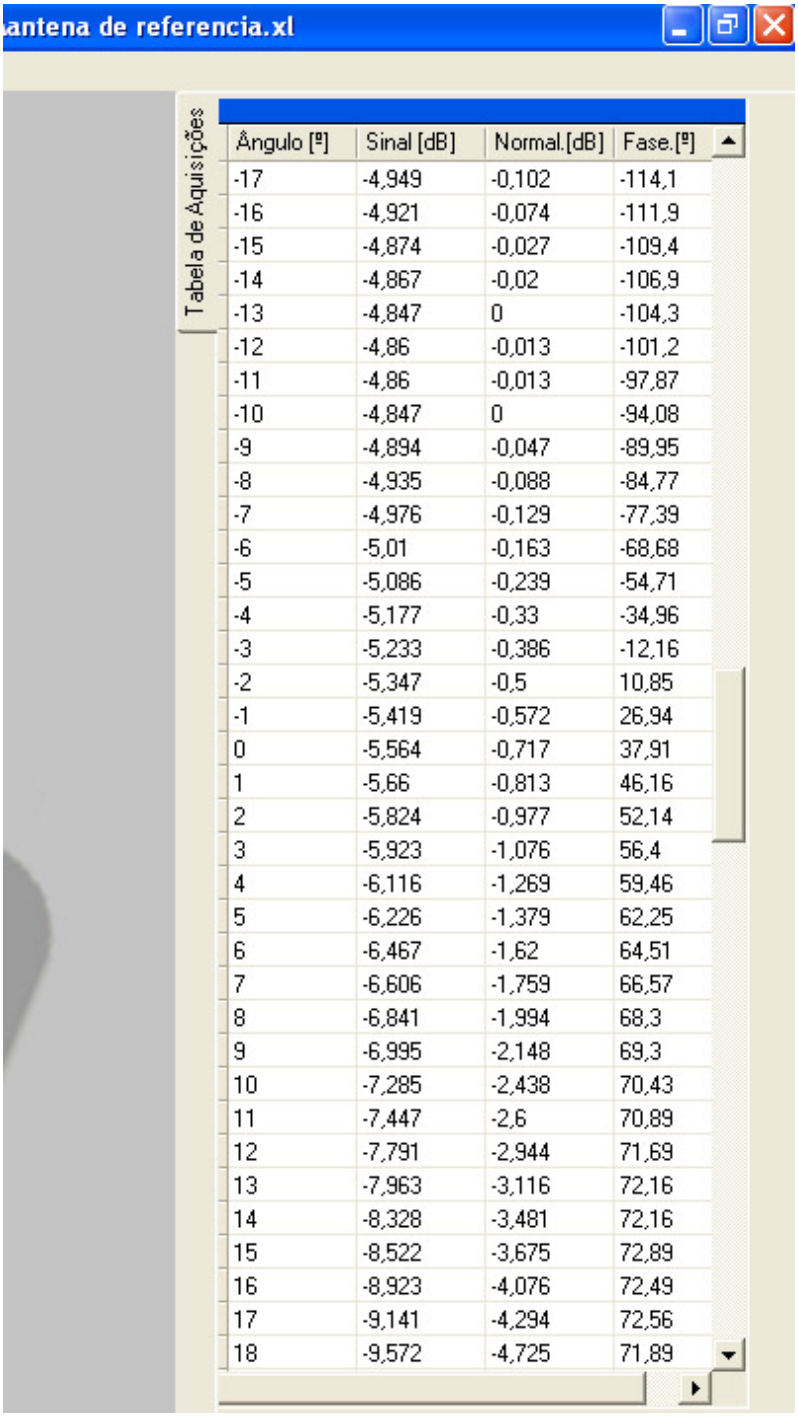


Figura 33 - Ilustração de tabelas, janelas e menus do RadScan.

3.3.2 Apresentação de dados

Os dados adquiridos pelo RadScan são colocados numa tabela após a aquisição de todos os valores, nesta tabela é possível editar valores para sua eventual correcção, (Figura 34), poderão ser gravados num ficheiro com a extensão .xl , ou exportados para o Excel (Figura 35). O Radscan permite ainda a compatibilidade de leitura com várias versões do software em MS-Dos permitindo que estes valores sejam carregados no novo software e gravados no formato mais actual. Os ficheiros produzidos pelo RadScan com extensão .xl, são ficheiros de texto, os quais poderão ser editados com qualquer editor de texto ou

importados para o Excel. São compostos por quatro colunas: ângulo [º]; valor medido [dB]; valor normalizado [dB] e fase [º].



Ângulo [º]	Sinal [dB]	Normal.[dB]	Fase.[º]
-17	-4,949	-0,102	-114,1
-16	-4,921	-0,074	-111,9
-15	-4,874	-0,027	-109,4
-14	-4,867	-0,02	-106,9
-13	-4,847	0	-104,3
-12	-4,86	-0,013	-101,2
-11	-4,86	-0,013	-97,87
-10	-4,847	0	-94,08
-9	-4,894	-0,047	-89,95
-8	-4,935	-0,088	-84,77
-7	-4,976	-0,129	-77,39
-6	-5,01	-0,163	-68,68
-5	-5,086	-0,239	-54,71
-4	-5,177	-0,33	-34,96
-3	-5,233	-0,386	-12,16
-2	-5,347	-0,5	10,85
-1	-5,419	-0,572	26,94
0	-5,564	-0,717	37,91
1	-5,66	-0,813	46,16
2	-5,824	-0,977	52,14
3	-5,923	-1,076	56,4
4	-6,116	-1,269	59,46
5	-6,226	-1,379	62,25
6	-6,467	-1,62	64,51
7	-6,606	-1,759	66,57
8	-6,841	-1,994	68,3
9	-6,995	-2,148	69,3
10	-7,285	-2,438	70,43
11	-7,447	-2,6	70,89
12	-7,791	-2,944	71,69
13	-7,963	-3,116	72,16
14	-8,328	-3,481	72,16
15	-8,522	-3,675	72,89
16	-8,923	-4,076	72,49
17	-9,141	-4,294	72,56
18	-9,572	-4,725	71,89

Figura 34 - Tabela de dados adquiridos.

A compatibilidade com o Excel é assim optimizada pela exportação directa dos valores e construção dos respectivos gráficos. Deste modo, e como já foi

brevemente referido, quando se usa a função de exportar para o Excel, o *software* abre uma folha já com os valores e respectivo gráfico em coordenadas lineares (Figura 35). O utilizador tem então oportunidade de trabalhar livremente, quer a nível das tabelas de valores, quer dos gráficos. Pode desta forma ajustar escalas e cores e/ou utilizar as restantes opções e funcionalidades do Excel.

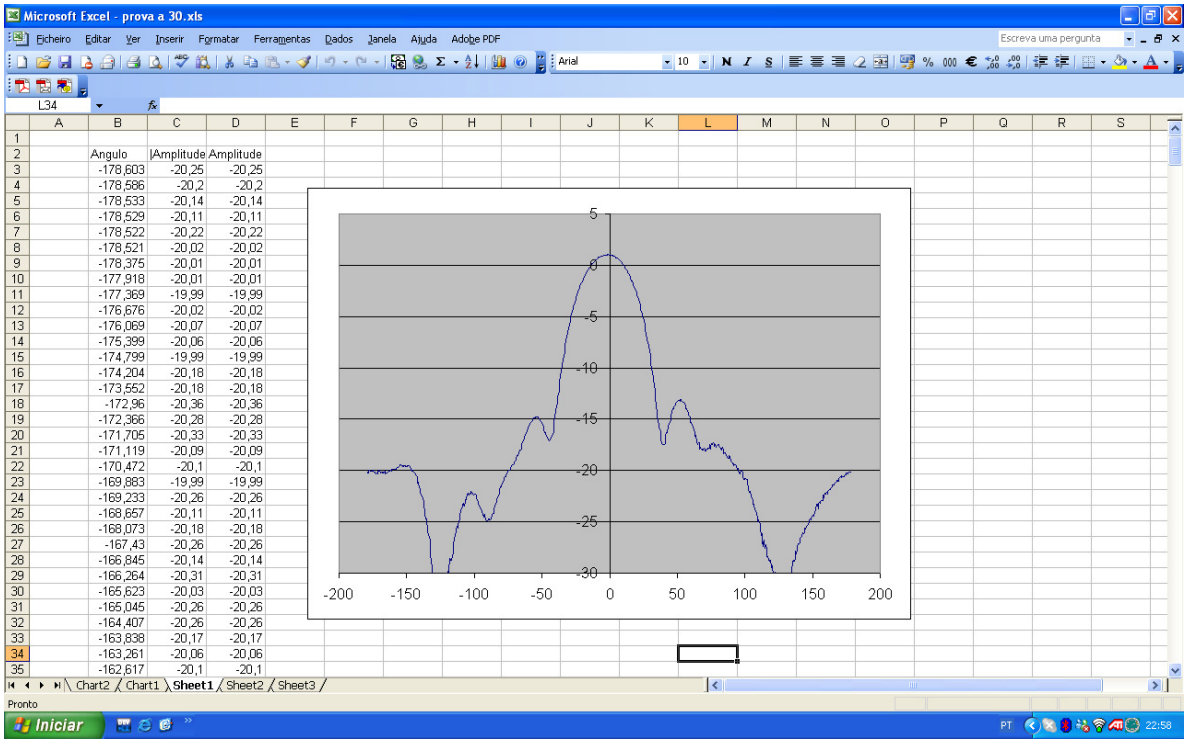


Figura 35 - Exemplo: Resultado de “Exportar Excel”.

4 Testes e Resultados

4.1 Resultados do estudo de usabilidade

Este *software* propõe-se realizar de uma forma mais simples e mais rápida as funções que o integram, dando resposta aos problemas mais prementes da usabilidade do sistema anterior (MS-DOS).

Pela análise da observação que consta do teste de usabilidade, comprova-se que o tempo de execução de tarefas para um utilizador inexperiente no sistema diminui consideravelmente. Mesmo um utilizador inexperiente utiliza de forma intuitiva o *software.* A gama de possíveis utilizadores torna-se assim mais vasta, pois estes conseguem completar as tarefas de forma autónoma e sem cometer erros na execução.

Desenvolveu-se portanto um *software* de fácil utilização, versátil e totalmente enquadrado no âmbito das aplicações e do sistema operativo actualmente mais utilizados. Versão de demonstração no Anexo E.

4.2 Fiabilidade de valores

Com a finalidade de averiguar a fiabilidade do RadScan foram realizadas várias medidas de diagramas de radiação de algumas antenas, com o objectivo de verificar em diversas situações se o comportamento do RadScan seria o esperado, o que se verificou para todos os casos. Não se apresentam aqui os diversos diagramas medidos, dado que objectivo desta dissertação não é o estudo do modo como as antenas radiam, mas sim do processo de medida de diagramas de radiação em geral.

No entanto apresenta-se em seguida, a título ilustrativo, os diagramas de radiação (em coordenadas lineares) correspondente a uma antena Yagi obtidos

usando os dois *softwares*, versão MS-DOS e o RadScan[16], para efeitos comparativos (Figura 36). Este tipo de gráfico apresenta os valores normalizados da amplitude de sinal recebido (em dB) em função da posição angular e está representado em duas escalas lineares, temos então o resultado a que se destina a aquisição de valores levada a cabo por este sistema de aquisição de dados, um diagrama de radiação. Estes diagramas têm por finalidade perceber o comportamento de uma antena em relação a sua orientação. Deste gráfico podemos observar a título de exemplo que esta antena terá a máxima capacidade de recepção/transmissão (lobo principal) numa faixa angular compreendida entre -40º e 30º em relação ao seu eixo. Também se pode observar que o máximo absoluto de sinal não está coincidente com o a direcção 0º definida neste digrama de radiação, o que poderá significar que ou a antena não foi previamente alinhada segundo o seu eixo físico, ou que o seu máximo de radiação não coincide com o seu eixo.

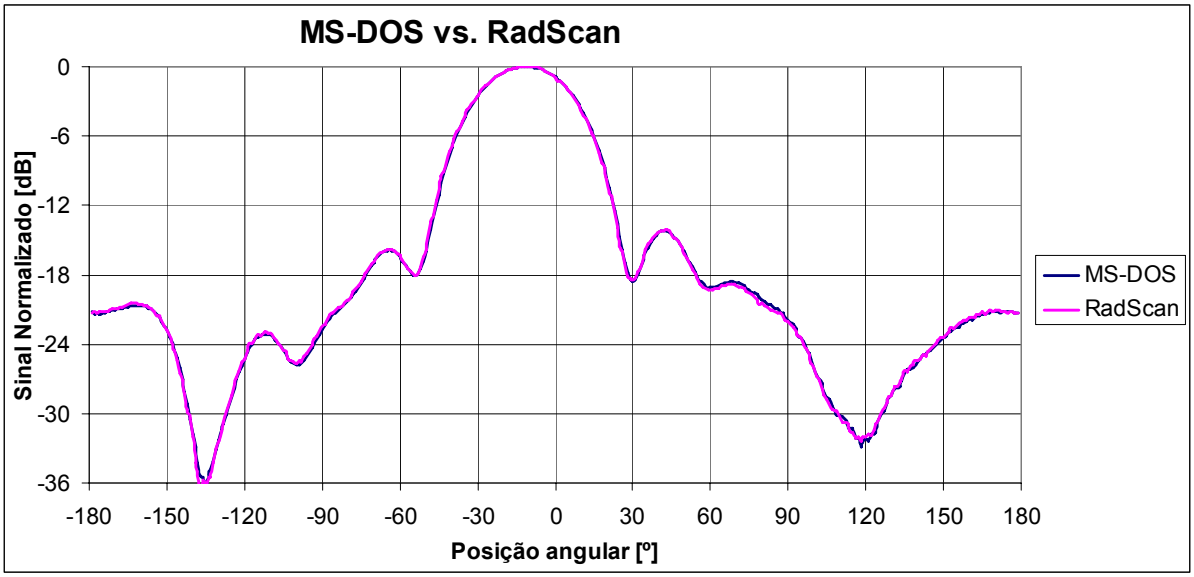


Figura 36 - Diagrama de radiação de uma antena Yagi, obtido aquisição por dois *softwares* distintos.

Como se pode verificar, as curvas dos valores de amplitude das duas aquisições são praticamente coincidentes. No entanto, antes de se retirarem

[16] Apresenta-se aqui apenas um diagrama de radiação embora este sistema já tenha sido testado na medição de várias antenas com bom resultados.

conclusões, observemos com mais detalhe um pequeno troço da amostragem acima ilustrada.

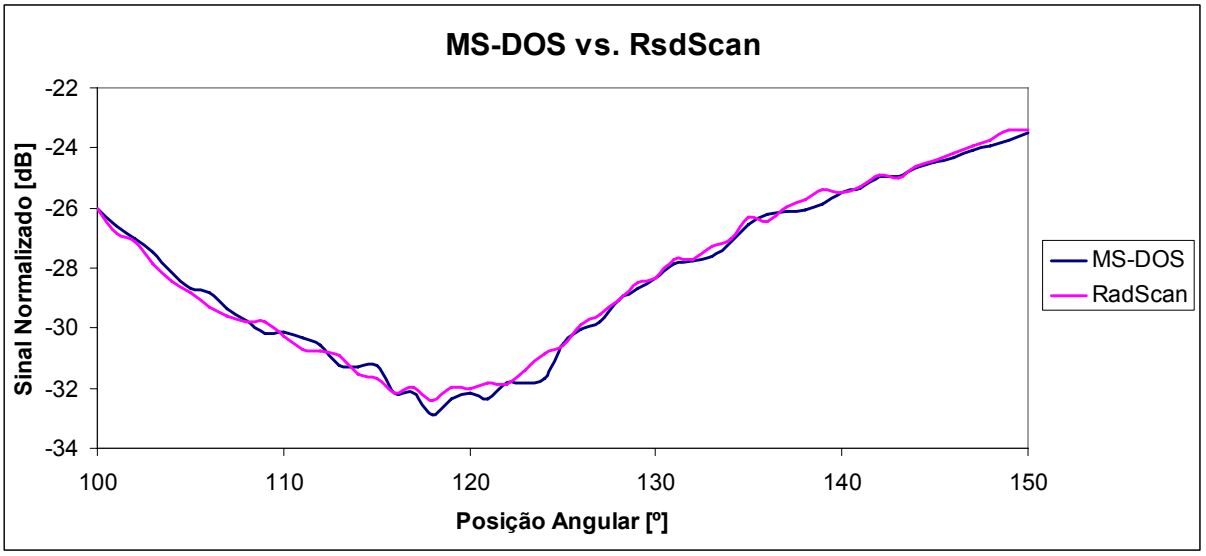


Figura 37 - Gráfico ampliado de um troço do diagrama de radiação da antena Yagi.

Neste gráfico, pode ver-se ampliada uma zona de maior *ripple*. As pequenas diferenças que podem ser observadas explicam-se sobretudo devido ao baixo nível dos sinais medidos.

Desta observação, pode retirar-se que o novo software apesar dos processos de aquisição seguirem percursos diferentes, apresenta resultados idênticos a aplicação desenvolvida em MS-DOS.

4.3 Performance do sistema

Através de análise experimental, verificou-se que usando as velocidades máximas permitidas pelo anterior software de MS-DOS, nem sempre se conseguiram as amostras necessárias para a realização correcta do diagrama. A solução passou por reduzir o limite da velocidade em 10%, o suficiente para garantir que o *software* consegue o mínimo de amostras necessárias por unidade de incremento da posição angular. A redução da velocidade diminui a probabilidade de falhas na aquisição, visto aumentar o número de amostragens, no entanto penaliza um pouco o tempo de aquisição. A título de exemplo, numa aquisição durante um arco de 358º, teremos 492 amostras a 40% de velocidade e 630 amostras a 30% de velocidade.

Contudo 10% de aumento da duração da tarefa de aquisição de dados, não é muito relevante, e não se nota num sistema que globalmente apresenta uma maior velocidade de utilização.

5 Especificações Técnicas

5.1.1 Descrição do *hardware* envolvido no sistema:

Câmara anecóica

Câmara de madeira revestida com materiais absorventes de radiações electromagnéticas, que permite um afastamento entre antenas de oito metros.

Controladora GPIB:

AT-GPIB/TNT

(Placa ISA para PC)

Hardware de Controlo Periférico:

Orbit AL-4901-3A *Position Controller* [2]

(Controlador do posicionador de azimute AL-360, GPIB)

Hardware de Medida:

Vector Voltmeter 8508A (Hewlett Packard) [3]

(Voltímetro vectorial, dispositivo de medida tensão e fase, GPIB)

Hardware de geração de sinal RF (exemplos):

Marconi Instrumentes 10KHz –1000MHz Signal Generator 2022

Wavetek - Synthesized Sinal Generator 2520A.

Acoplador direccional:

HP-7788-012

5.1.2 Requisitos do *software* RadScan

Requisitos para executar o RadScan:

- Computador:
  Intel PentiumIII – 730MHz, RAM: 128M, Disco 20G (requisitos mínimos).
- Sistema operativo
  Windows XP
- Microsoft framework 1.1
- *Drivers* da controladora GPIB,
  Apenas fornecido por pedido à National mediante número de séria da placa, visto ser uma versão muito antiga. (Disponível no CD, Anexo E. - Código)
- Office 2003 (Excel)
- Acrobat Reader / Writer

5.1.3 Características Técnicas do sistema

Nesta secção descrevem-se as características técnicas globais do sistema, resultantes das limitações ou capacidades de cada um dos equipamentos. Estas características são as seguintes:

Ângulo extremos de rotação: -179º e 179º, limitação por *software*.

Incrementos angulares possíveis: 1, 2, 3, 5, 10º.

Precisão do movimento:

Erro de posicionamento: 0.01º

Erro de leitura de posição: 0.001º

Velocidade máxima a 99%: 17º/s

Velocidades aconselháveis em função do incremento angular
Assumindo que apenas está em curso a aplicação RadScan

Incremento[º]	Velocidade[%]
1	30
2	70
3	80
5	99
10	99

Tabela 4 - Incremento possível face a velocidade

Atraso médio de leitura (tempo decorrido desde a obtenção do valor da posição até obtenção do valor da amplitude e fase): 50ms

Conclusão

O presente trabalho partiu da necessidade de alterar o interface com o utilizador, do sistema de mediação de antenas de diagramas de radiação de antenas que estava a ser usado. De facto, vieram a revelar-se lacunas profundas aquando do teste desse sistema. Dada a evolução da informática e dos meios hoje em dia disponíveis considerou-se ser possível realizar um novo software, que desse resposta às necessidades mais prementes dos utilizadores.

A avaliação de usabilidade do software consistiu numa primeira fase numa avaliação heurística, à qual se seguiu o teste de usabilidade e a observação de utilizadores em ambiente real. Os resultados desta avaliação demonstraram quão fácil é a utilização do novo *software* face ao anterior. Deste modo, não restaram dúvidas quanto à mais-valia que o RadScan vem trazer na utilização do sistema de medição de antenas, dada a facilidade de utilização do seu interface.

C# foi seleccionada como linguagem de programação do RadScan por ser uma das linguagens mais recentes e considerada linguagem de referência por muitos programadores. Esta mostrou-se à altura das exigências do desenvolvimento do *software.* A evolução de sistemas tem um ritmo muito acelerado, é difícil prever ainda se C# prevalecerá como uma linguagem de referência. Deste ponto de vista, será necessário um acompanhamento do sistema de controlo às novas tecnologias tanto ao nível do *hardware* como do *software.* A linguagem C# poderá acompanhar alguma desta evolução à medida que forem surgindo novas bibliotecas de funções e tutoriais.

Face aos equipamentos disponíveis no sistema de medição de antenas, o GPIB foi o protocolo de comunicação utilizado, visto os equipamentos datarem de uma altura em que o GPIB era o mais rápido e robusto. É de ressalvar que à medida que forem sendo introduzidos novos equipamentos, haverá a necessidade de introduzir também a comunicação por USB, comunicação dominante hoje em dia.[17]

Ainda quanto à comunicação entre equipamentos GPIB, o *software* "*Max - National Instruments Measurement & Automation Explorer*" trouxe valiosas ferramentas para a construção de toda a rede de comunicação, mostrando ser um *software* muito versátil. Este *software* faz parte do *driver* da placa GPIB e facilita a interacção de aplicações com os dispositivos GPIB, assumindo o endereçamento de cada um deles.

Desta forma, aplicações como o RadScan precisam unicamente de ter a informação do dispositivo com que quer comunicar e aquela a enviar. A versão posterior ao *software* "*Max - National Instruments Measurement & Automation Explorer*" usado no RadScan, permitiria inclusive comunicar com dispositivos com protocolo GPIB e interface física, Universal Serial Bus (USB). Neste caso, o dispositivo seria incluído na lista de dispositivos do *software* do *driver* (*Max*) e abordado de forma semelhante aos anteriores. No entanto, a placa controladora GPIB existente no PC não permitiu tal actualização.

O CommLink mostrou-se igualmente importante no desenrolar eficiente do processo de comunicação com o posicionador e na selecção de entre os vários tipos de movimento disponibilizados pelo mesmo posicionador. Este mostrou-se um equipamento extremamente eficaz quanto às sequências e precisão de movimentos, entre os quais se encontram os mais adequados a cada diferente etapa do procedimento de aquisição de dados. Como se trata de um posicionador algo antigo, implica uma grande complexidade na sua comunicação, visto utilizar a primeira versão do protocolo GPIB e os comandos específicos do fabricante.

Por estas razões, procedeu-se com grande cuidado à criação de funções de comunicação com os dispositivos, pois estas poderão ser reaproveitadas em aplicações futuras. Além disso, constituem peças-chave e recorrentes do

[17] Esta introdução exigirá um *upgrade* da Placa GPIB do PC ou em alternativa, código dedicado a tratar directamente cada dispositivo inserido.

RadScan. O maior número destas funções de baixo nível é dedicado à comunicação com o posicionador que, dado o seu protocolo básico, exige muito processamento de informação.

Um outro aspecto também desenvolvido em relação à anterior versão, foi a compatibilidade de ficheiros com o Excel, implementando-se a importação directa de dados e a criação de gráficos de radiação.

Verificou-se que os diagramas de radiação obtidos através do novo software são idênticos aos obtidos na versão anterior do sistema de medição. Assim, obteve-se uma aplicação que globalmente é mais fácil de usar (*user friendly*) e torna a interacção com o utilizador mais rápida e prática.

O RadScan poderá ainda ser optimizado. O trabalho, a ser desenvolvido futuramente, poderá abarcar vários aspectos. Entre estes, é de destacar a introdução de um Offset automático, ou seja, o RadScan procuraria de forma autónoma a posição correspondente ao máximo de amplitude do sinal. Um outro aspecto a desenvolver, seria o melhoramento da função “Ajuda”, introduzindo o Manual de Utilizador. A geração de um diagrama em coordenadas polares, poderia ser profícua. Por último, refira-se que poder-se-ia adicionar o controlo de geradores de radiofrequência (RF) de forma a poder programar o RadScan para realizar aquisição de dados da mesma antena por varrimento automático em frequência ou amplitude.

Este *software* necessitará de contínuo desenvolvimento com a introdução de novos equipamentos e funcionalidades, sendo de evidenciar a importância da interface com o utilizador e o cuidado que se deverá ter nas questões de usabilidade no futuro.

Referências

[1] - Pereira, Prof. José Fernando da Rocha, Proposta de Projecto: "Desenvolvimento do *software* de comando dum sistema de medição de antenas.".

[2] - Orbit Advanced Technologies "Communication link for AL-4706-3A", Agosto de 1991

[3] - Hewlett-Packard, "Remote Operation. The Hewlett-Packard interface Bus"

[4] - Malaquias, Paulo J. M., "Relatório de Projecto – *Software* para teste de Antenas", Setembro 1992 e Agosto de 2003, Universidade de Aveiro – Departamento de Electrónica

[5] - Sharp, John, Jagger, Jon, "Microsoft Visual C# .Net step by step" 2003

[6] - msdn library help of Visual Studio.Net 2003

[7] - Nielsen, J., "Ten usability heuristics", http://www.useit.com/papers/heuristic/heuristic_list.html (visitado em Janeiro 2007).

[8] - "Microsoft Visual Estúdio C# Developer Center" http://msdn.microsoft.com/vcsharp/programming/ (última consulta Janeiro 2007)

[9]- Fisher, Eugene, Jersen, C.W., "Pet and the IEEE488 Bus (GPIB)", McGraw-Hill 1984

[10]- National Intruments, Visual Studio .NET http://zone.ni.com/devzone/devzone.nsf/webcategories/0139F72E7AEF769586256B6500567033 (última consulta Janeiro 2007)

[11] - Orbit Advanced Technologies, "AL-4901-3A Controller with Built-In CPU Operation Manual", Agosto de 1991

[12] - National Intruments, NI-488.2 User Manual..

[13] - Nielsen, J., “Ten usability heuristics”, http://www.useit.com/papers/heuristic/heuristic_list.html (visitado em Janeiro 2007).

[14] - Electronics Group, University Amsterdam, Gpib programming tutorial.

[15] - Luís Pereira, Universidade de Aveiro, Instrumentação e Técnicas de Medida Instrumentação Electrónica para Física 2005/2006.

[16] - National Instruments, GPIB Tutorial, http://www.natinst.com (National Instruments, última consulta Janeiro 2007)

[17] - Santos, Beatriz S., “Apontamentos da disciplina de Interface Humano Computador 2005/6.”, Universidade de Aveiro

Anexo A. - Heurísticas de Usabilidade de Jakob Nielsen

1. Visibilidade do estado do sistema

   O utilizador deve estar sempre informado sobre o que se passa durante a sua interacção com o sistema. Esta informação deve ser apresentada em tempo útil seguindo um mecanismo de *feedback*.

2. Correspondência entre o sistema e o mundo real

   O sistema deve comunicar com a mesma linguagem do utilizador, através de palavras, frases e conceitos familiares, em vez de termos orientados ao sistema, seguindo convenções do mundo real, fazendo aparecer a informação de uma maneira natural e lógica.

3. Controlo e liberdade por parte do utilizador

   Geralmente, os utilizadores escolhem funções de sistema por engano e necessitam de saídas de emergência claramente distinguíveis de modo a não terem que passar por diálogos extensos. Suporte de mecanismos tipo “undo” e “redo”.

4. Consistência e Padrões

   O utilizador não deve questionar-se se diferentes palavras, acções ou situações significam o mesmo. Deve-se evitar ambiguidades e seguir as convenções da plataforma.

5. Prevenção de Erros

   Melhor do que apresentar boas e explícitas mensagens de erro, é realizar uma interface cuidada, de modo a evitar que esses erros ocorram.

6. Reconhecer em vez de recordar

   Criar objectos, acções e opções visíveis. O utilizador não deve ser obrigado a lembrar-se da informação de diálogos anteriores. As instruções para o uso do sistema devem ser visíveis ou facilmente acessíveis quando necessárias.

7. Uso flexível e eficaz

Aceleradores, invisíveis ao utilizador inexperiente, podem acelerar a interacção de utilizadores experientes, podendo o sistema adaptar-se ao grau de experiência dos utilizadores. O sistema deve permitir que os utilizadores possam configurar o sistema para acções frequentes.

8. Design estético e minimalista

Os diálogos não devem conter informação irrelevante ou desnecessária. Cada unidade extra de informação num diálogo compete com as unidades relevantes de informação, diminuindo a sua visibilidade relativa.

9. Ajuda no reconhecimento, diagnóstico e recuperação de erros

Mensagens de erro devem ser expressas em linguagem clara (sem códigos), indicando o problema com precisão e sugerindo uma solução construtiva.

10. Ajuda e Documentação

Apesar de ser possível que o sistema seja usado sem recorrer a qualquer documentação ou ajuda, esta deve ser facultada. Qualquer ajuda ou informação deve ser facilmente acessível, orientada para a tarefa actual do utilizador, listada com os passos a efectuar e não ser demasiado longa.

Anexo B. - Classificação de erros

0 - Não é um problema de usabilidade;

1- Apenas problema de estética: não requer resolução imediata, a não ser que haja tempo extra no projecto;

2 - Problema de usabilidade menor: deve ser dada baixa prioridade à sua resolução;

3 - Problema de usabilidade maior: deve ser dada alta prioridade à sua resolução;

4 - Problema de usabilidade catastrófica: é imperativa a sua resolução.

Anexo C. - GPIB sinais e linhas

1. Linhas GPIB

O cabo de conexão GPIB é constituído por 16 linhas de sinal e 8 sinais de massa. As 16 linhas de sinal agrupam-se em 8 linhas de dados, 3 linhas de *handshake* e 5 linhas de controlo (Figura 38).

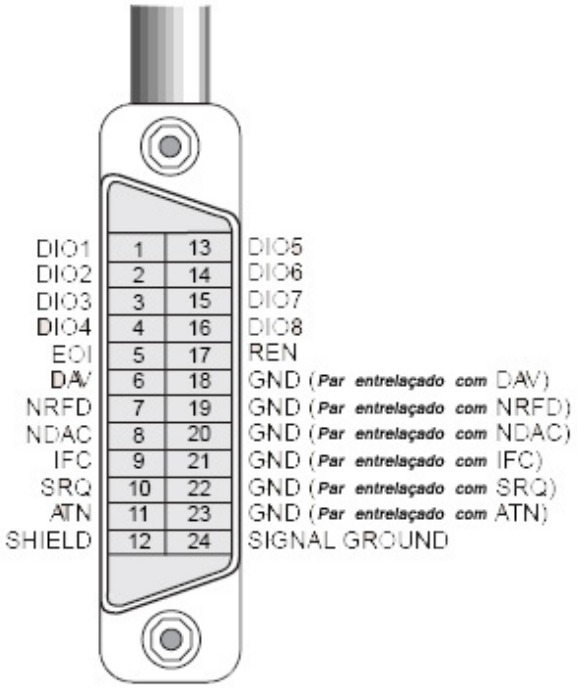


Figura 38 - Ficha GPIB, respectivas linhas de comunicação.

Linhas de Informação

As 8 linhas desde DIO1 até DIO8, transportam quer mensagens de dados como mensagens de instruções. O estado de *Attention* (ATN) determina se a informação transmitida é dados ou instruções. Os comandos e maior parte das instruções usam sistema ASCII ou ISO, 7 bits, em que o oitavo bit (DIO8) não é usado ou então é usado para paridade.

Linhas de *Handshake*

Três linhas controlam de forma assíncrona a transferência de mensagens entre os dispositivos. Este processo, denominado de *3-wire interlocked handshake,* garante que os bytes das mensagens no barramento de dados são entregues e recebidos sem erros de transmissão.
As 3 linhas são compostas por:

-NRFD (*not ready for data*) – Determina quando o dispositivo está pronto ou não para a recepção do byte de mensagem. A linha é acedida por todos os dispositivos, por *listeners* quando se recebem mensagens de dados e pelo *talker* quando activa o protocolo 488.

-NDAC (*not data accepted*) – Indica se o dispositivo aceitou ou não o byte de mensagem. A linha é acedida por todos os dispositivos quando se recebem comandos e por *listeners* quando recebem mensagens de dados.

- DAV (*data valid*) – Assinala quando a informação no bus está valida (estável) e poderá ser lida de forma segura pelos dispositivos. O controlador acede a linha DAV quando envia comandos e o *talker* acede quando envia mensagens de dados.

Linhas de interface e controlo

Cinco linhas que controlam o fluxo de mensagens:

- ATN (*Attention*) – O controlador activa a linha ATN quando usa o bus de dados para enviar comandos. Para o nível baixo significa que o *Talker* pode enviar mensagens.
- IFC (*Interface Clear*) – O controlador do sistema usa a linha IFC para inicializar o bus e assumir a posição de CIC.
- REN (*Remote Enable*) – O controlador usa a linha REN para determinar o modo de programação dos dispositivos, remoto ou local.
- SRQ (*Service request*) - Qualquer dispositivo poderá usar esta linha para de forma assíncrona requerer serviços ao controlador.
- EOI (*End of identif*") – Esta linha tem dois propósitos. O *talker* usa-a para assinalar o final de uma mensagem (*string*), o controlador usa a linha EOI para que os dispositivos identifiquem a sua resposta durante o *poling* paralelo.

2. Características físicas e Eléctricas

Os Dispositivos são normalmente interligados por um cabo de 24 fios condutores blindados, com uma ficha igual em ambas as extremidades contendo numa das faces da ficha um encaixe macho e na face oposta um encaixe fêmea (Figura 39).

Figura 39 - Cabo GPIB (cada extremidade contém duas fichas de ligação macho e fêmea)

Os dispositivos podem ser ligados na configuração linear ou numa configuração estrela (Figura 40) ou ainda numa combinação de ambos. O tipo de conector é do tipo: Amphenol ou Cinch Séries 57 MICROIBON ou AMPCHAMP.
O GPIB usa lógica negativa com níveis standards de TTL, nível baixo menor ou igual a 0,8V e nível alto maior ou igual a 2V.

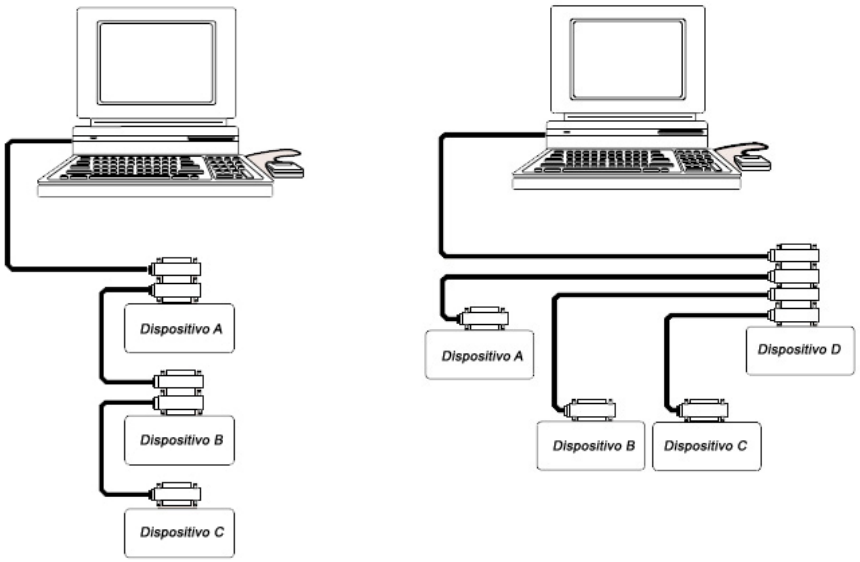


Figura 40 - Configuração de ligações entre dispositivos, linear a esquerda e estrela a direita.

3. Requerimentos de configuração

Para atingir as altas taxas de transferência, para as quais o GPIB foi concebido, o comprimento dos cabos e o número de dispositivos é limitado.

- Restrições para funcionamento normal:

  - Máximo de separação de 4m entre dois dispositivos e uma média de distância de 2m em todo o bus.

  - Máximo comprimento total do cabo de 20m.

  - Um máximo de 15 dispositivos ligados ao barramento e não menos do que um terço em estado de ligados.

- Para sistemas de alta velocidade:
  - Um máximo de 15m de cabo com um dispositivo por metro de cabo.
  - Todos os dispositivos deverão estar em estado de ligados.
  - Todos os dispositivos deverão usar portas de três estados de 48mA.
  - A capacidade em cada linha do GPIB deverá ser inferior a 50pF por dispositivo.

4. IEEE 488.2 e SCPI

As normas SPCI e IEEE 488.2 desenvolveram-se no sentido de eliminar as limitações e ambiguidades da norma original IEEE 488.

A norma IEEE 488.2 torna possível o engenho de sistemas de teste mais compatíveis e produtivos. O sistema SCPI simplifica a tarefa de programação ao definir uma só estrutura de comandos compreensiva para instrumentos programáveis, independentemente do fabricante. Na Figura 41 pode observar-se a abrangência de cada uma das normas IEEE 486, IEEE 486.2, SCPI.

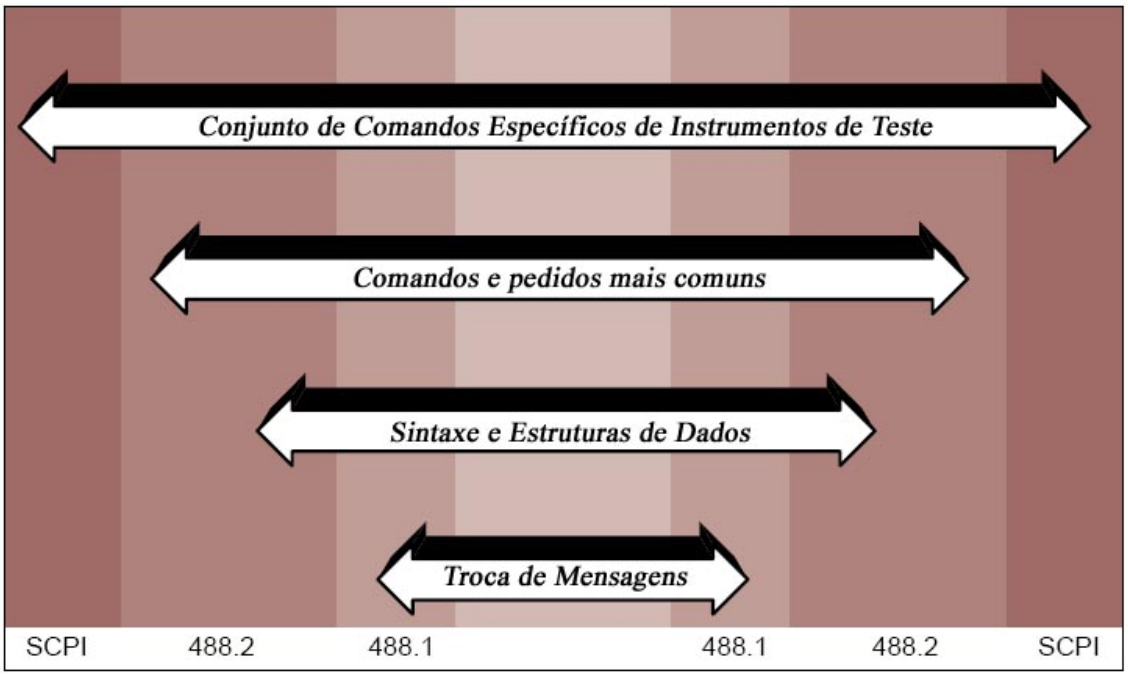


Figura 41 - Evolução das normas GPIB.

A norma ANSI/IEEE 488-1975, agora denominado de IEEE 488.1 simplificou largamente a inter conexão de instrumentação programável ao definir especificações mecânicas, eléctricas e de protocolos de hardware. Pela primeira vez, instrumentos de diferentes fabricantes foram interligados por um cabo standard. Embora esta norma tenha percorrido uma longa caminhada e muitos percalços no que toca ao desenvolvimento de produtividade dos engenheiros de teste. Especificamente, o sistema IEEE 488.1 não endereçava formatos de dados, relatório de estado, protocolo de troca de mensagens, comandos de configuração comuns ou comandos específicos ao dispositivo. Como resultado de tal, cada fabricante implementou estes itens de forma desigual, deixando a pessoa que desenvolve os sistemas de teste com uma tarefa de veras elaborada.

O sistema IEEE 488.2 melhorou e fortaleceu a norma IEEE 488.1 ao normalizar também os formatos de dados, relatório de estado, controlo de erros, funcionalidade dos controlos e comandos comuns, a que todos os instrumentos têm de responder de forma definida. Com estas normalizações os sistemas IEEE488.2 tornam-se mais compatíveis e fiáveis. A norma IEEE 488.2 foca primariamente o protocolo de software, mantendo assim a compatibilidade com a norma IEEE 488.1 orientada para o hardware. A SPCI construída sobre a norma IEEE488.2 define comandos específicos que normalizam a programação de instrumentos. Os sistemas SCPI são muito mais fáceis de programar e de manter. Em muitos casos, é possível fazer intercambio ou actualização de instrumentos sem ser necessário mudar de programa de teste. A combinação de SCPI e IEEE488.2 oferece ganhos de produtividades significativos e finalmente oferece uma normalização de software assim como a norma IEEE488.2 ofereceu uma normalização de hardware.

5. IEEE 488.2

A norma IEEE488.2 de 1987 encorajou um novo nível de crescimento e aceitação do bus IEEE 488.2 ou GPIB, tomando em conta de problemas que se levantaram na versão anterior IEEE488. A IEEE488.2 foi idealizada na premissa de ficar compatível com a norma IEEE488. O conceito de *overreing* usado na especificação IEEE488.2 para a comunicação entre controladores e instrumentos

faz uso de uma *precise talking* e *forgiving listening*, por outras palavras a IEEE488.2 definiu exactamente como os controladores e instrumentos comunicam, para que se obtenha um sistema de confiança, eficiente, compatível com IEEE488.2. A norma também exige que os dispositivos deste sistema estejam aptos a funcionar com os dispositivos IEEE488.1 através da aceitação de uma vasta gama de comandos e formatos de dados como *listener*. Pode-se obter verdadeiros benefícios da norma IEEE488.2 quando se tem um sistema verdadeiramente compatível.

6. Controladores

Embora a norma IEEE488.2 tenha tido menos impacto nos controladores do que teve nos instrumentos, há certos requerimentos e melhorias para controladores que fizeram um controlador IEEE488.2 ser um componente de teste de sistemas necessário. A norma IEEE488.2 definiu com precisão a maneira como os controladores enviam comandos e informação e ainda algumas funcionalidades acrescidas. Devido a estes requisitos do controlador, os fabricantes dos instrumentos podem idealizar mais instrumentos compatíveis e eficientes. Os benefícios desta normalização são a redução tempo e custo de desenvolvimento, pois resolve o problema causado pelas incompatibilidades dos instrumentos a variação das estruturas de comando e os formatos de dados.

7. Requisitos dos controladores IEEE488.2

A norma IEEE488.2 definiu requisitos para os controladores, incluindo um conjunto de capacidades a ser usadas na interface IEEE488.1, tais como:

- Envio de impulsos no bus GPIB durante 100us quando este não tem sinal,
- Detecção ou especificação do fim de linha,
- Activar ou desactivar a linha de REN,
- Detecção do estado de transição da linha SRQ,
- Detecção do estado da NDAC,
- Libertação da linha ao fim de um determinado tempo numa transacção de I/O.

Outros requisitos chave para os controladores são sequências de controlo de *bus* e protocolos de *bus*.

8. Controlo de sequências IEEE 488.2

As sequências de controlo definidas na norma IEEE 488.2, especificam as mensagens IEEE488.1 que são enviados do controlador, bem como pedidos de múltiplas mensagens. A norma IEEE488.2 definiu 15 sequências de controlo necessárias e 4 sequências de controlo opcionais (Tabela 5). As sequências IEEE488.2 descrevem os estados do GPIB e fazem o pedido de mensagens de comando para cada uma das operações defendias. As sequências de controlo IEEE488.2 removem a ambiguidade de possíveis condições de bus. Assim os instrumentos e controladores são muito mais compatíveis. Ao definir exactamente o estado do bus e como os dispositivos devem responder a mensagens específicas a IEEE488.2 resolve problemas de desenvolvimento de sistemas.

Description	Control Sequence	Compliance
Send ATN-true commands	SEND COMMAND	Mandatory
Set address to send data	SEND SETUP	Mandatory
Send ATN-false data	SEND DATA BYTES	Mandatory
Send a program message	SEND	Mandatory
Set address to receive data	RECEIVE SETUP	Mandatory
Receive ATN-false data	RECEIVE RESPONSE MESSAGE	Mandatory
Receive a response message	RECEIVE	Mandatory
Pulse IFC line	SEND IFC	Mandatory
Place devices in DCAS	DEVICE CLEAR	Mandatory
Place devices in local state	ENABLE LOCAL CONTROLS	Mandatory
Place devices in remote state	ENABLE REMOTE	Mandatory
Place devices in remote with local lockout state	SET RWLS	Mandatory
Place devices in local lockout state	SEND LLO	Mandatory
Read IEEE 488.1 status byte	READ STATUS BYTE	Mandatory
Send group execution trigger (GET) message	TRIGGER	Mandatory
Give control to another device	PASS CONTROL	Optional
Conduct a parallel poll	PERFORM PARALLEL POLL	Optional
Configure devices' parallel poll responses	PARALLEL POLL CONFIGURE	Optional
Disable devices' parallel poll capability	PARALLEL POLL UNCONFIGURE	Optional

Tabela 5 - IEEE 488.2 *Required and Optional Control Sequences*

Key	Name	Compliance
RESET	Reset System	Mandatory
FINDRQS	Find Device Requesting Service	Optional
ALLSPOLL	Serial Poll All Devices	Mandatory
PASSCTL	Pass Control	Optional
REQUESTCTL	Request Control	Optional
FINDLSTN	Find Listeners	Optional
SETADD	Set Address	Optional, but requires FINDLSTN
TESTSYS	Self-Test System	Optional

Tabela 6 - IEEE 488.2 *Required and Optional Control Sequences*

9. Protocolos IEEE488.2

Os protocolos são rotinas de alto nível que combinam um número de sequências de controlo para realizar operações comuns de teste de sistemas. A norma IEEE488.2 define dois protocolos essenciais e seis opcionais (Tabela 6). Estes protocolos reduzem o tempo de desenvolvimento pois combinam vários comandos que executam operações mais comuns, essenciais a qualquer sistema de teste. O protocolo RESET assegura que o GPIB e todos os dispositivos foram inicializados e colocados num estado conhecido. O protocolo ALLSPOLL faz *polling* em série a cada dispositivo, retornando o byte de estado de cada um. O protocolo PASSCTL e REQUESTCTL comutam o controlo do bus entre dispositivos diferentes. O protocolo TESTSYS envia instruções a cada dispositivo para executar o seu auto teste, reportando de volta ao controlador com a informação de existência de problema ou se este está pronto a operar. Talvez os protocolos mais importantes sejam o FINDLSTN e o FINDRQS pois tiram vantagem das capacidades do controlador IEEE488.2 monitorizando as linhas de bus e localizando os dispositivos à escuta no bus. O controlador implementa o protocolo FINDLSTN percorrendo o espaço de endereçamento no bus e monitorizando a linha de *handshake* ANDDOC para determinar se algum dispositivo existe com esse endereço. O resultado do FINLSTN é uma lista de endereços de todos os dispositivos detectados. O FINDLSTN é usado no inicio de uma aplicação para assegurar a configuração adequada do sistema e para providenciar uma lista valida de dispositivos GPIB, que podem ser usados como

parâmetros de entrada por todos os outros protocolos IEEE488.2. A capacidade de monitorização da linha de bus de um controlador IEEE488.2 é também útil para detectar e diagnosticar problemas dentro de um sistema de teste.

O protocolo FINDRQS é um mecanismo eficiente para a localização e monitorização (por *polling)* de dispositivos que solicitam serviço. O controlador usa a sua sensibilidade a transição de falso para verdadeiro da linha SRQ. Configura-se a prioridade na lista de dispositivos de entrada, para que os dispositivos mais críticos recebam serviço em primeiro lugar. Se a aplicação poder comutar para este protocolo de imediato, aquando a inserção no sistema uma linha SRQ, aumentando-se a eficiência e capacidade de resposta (*throughput*) do mesmo.

10. Instrumentos IEEE488.2

As especificações da norma IEEE488.2 para instrumentos podem exigir mudanças no firmware e possivelmente no hardware. Contudo os instrumentos IEEE488.2 são mais fáceis de programar pois respondem a comandos comuns, usam consultas pedidos numa maneira definida usando protocolos normalizados de troca de mensagens e de formato de dados. A norma IEEE488.2 é a base da norma SPCI que torna a programação dos sistemas de teste ainda mais fácil. A IEEE488.2 define um conjunto mínimo de capacidades de interface da norma IEEE488.1 que um instrumento deve ter. Todos os dispositivos devem estar aptos a enviar e receber informação, requerer serviços e responder a uma mensagem de CLEAR de outro dispositivo. A norma IEEE488.2 define com precisão o formato de comandos a enviar aos instrumentos e o formato e codificação das respostas enviadas pelos instrumentos. Todos os instrumentos têm de realizar certas operações para comunicar no bus e obter informação de estado. Por estas operações serem comuns a todos os instrumentos, a norma IEEE488.2 definiu os comandos de programação usados para executar estas operações e as consultas usadas para receber informações de estado. Estes comandos comuns e consultas são mostrados na (Tabela 7). Com a normalização dos relatórios de estado com a norma IEEE488.2, o controlador sabe como obter informação de estado sobre cada dispositivo no sistema. O

modelo de entrega de relatório de estado é construído através do byte de estado do IEEE488.1, de forma a providenciar informação de estado mais detalhada. Este modelo pode ser visto na integra no (Diagrama 5).

Mnemonic	Group	Description
*IDN?	System Data	Identification query
*RST	Internal Operations	Reset
*TST?	Internal Operations	Self-test query
*OPC	Synchronization	Operation complete
*OPC?	Synchronization	Operation complete query
*WAI	Synchronization	Wait to complete
*CLS	Status and Event	Clear status
*ESE	Status and Event	Event status enable
*ESE?	Status and Event	Event status enable query
*ESR?	Status and Event	Event status register query
*SRE	Status and Event	Service request enable
*SRE?	Status and Event	Service request enable query
*STB?	Status and Event	Read status byte query

Tabela 7 - IEEE 488.2 *Mandatory Common Commands*

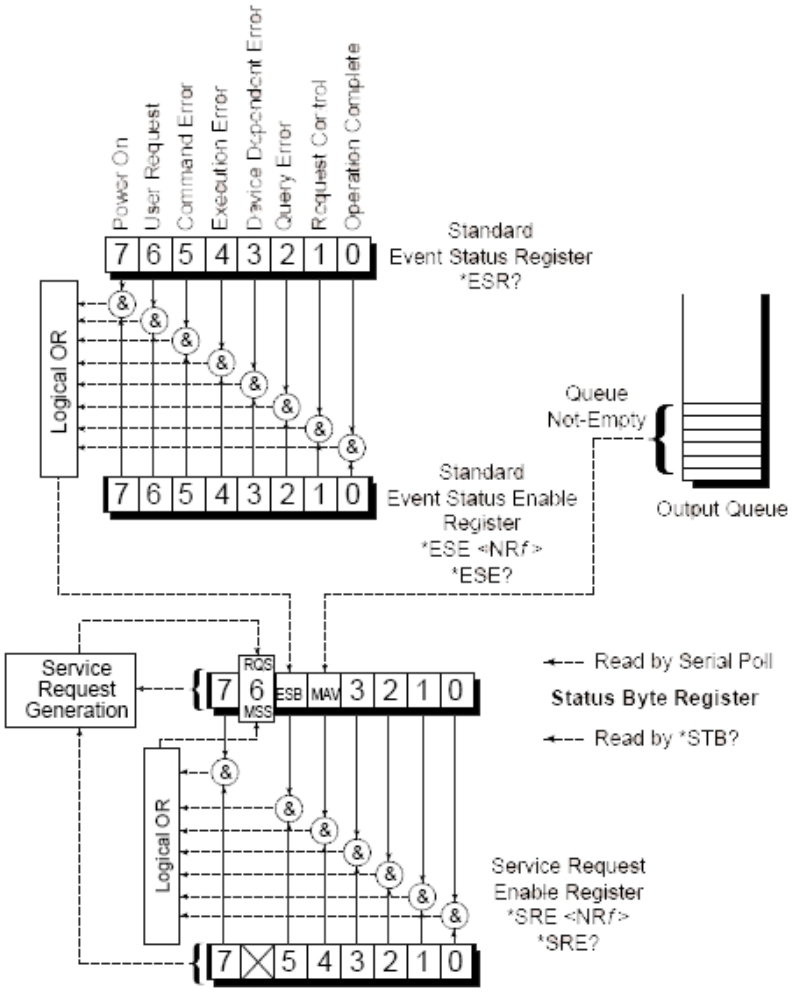


Diagrama 5 - *Status Report Model*

Anexo D. - Manual do utilizador

Anexo E. - Código

O código-fonte devido à sua extensão não foi incluído deste documento, encontra-se na íntegra no CD-Rom anexo. As funções de baixo nível estão descritas de forma resumida na secção 3.2 Funções de comunicação com os dispositivos.

Neste CD-Rom, temos também:

- O software de MS-DOS;
- A maqueta Inicial
- Uma versão de demonstração do RadScan
- A versão de instalação RadScan
- *Drivers* da placa AT-GPIB
- Manual do Controlador Orbit
- Manual do Utilizador
- A presente dissertação
- Commlink
- FrameWork 1.1
- Acrobat Reader